Use of Generative Learning to Improve Realism in Fluvial Facies Modelling

CHAO SUN

A thesis submitted in fulfilment of the requirements

for the degree of Doctor of Philosophy

Institute of GeoEnergy Engineering

School of Energy, Geoscience, Infrastructure and Society

Heriot-Watt University

June 2023

Abstract

This thesis investigates using generative adversarial networks (GANs) to fast-build geologically plausible 3D facies models of fluvial systems by learning simulated facies patterns and their uncertainty from a process-based model with different avulsion parameters. Fluvial systems, e.g. meandering rivers, can create complicated facies distributions composed of multiple facies with varied shapes and transitions due to the complex sedimentary processes and the partiality of the resulting record. Conventional simulation tools, such as process-based models and geostatistical approaches, use a stochastic process to simulate fluvial facies models based on physic-based or rule-based processes, parametric geometries or spatial correlation models. The stochastic nature allows those conventional tools to produce an ensemble of different realisations. However, those realisations often can't be directly sampled when integrated into a model updating loop, requiring external geological parametrisation, e.g. PCA. Deep generative models, e.g. GANs, showed powerful learning capability that allows using a small number of latent parameters obeying a simple distribution, e.g. Gaussian or Uniform distribution, to sample random realisations, which can be regarded as a geological parameterisation itself. GANs have successfully reproduced realisations from object-based models. This triggers the interest in exploiting the learning capacity for data complexity, capturing geological processes more closely. As GANs can learn geological patterns from object-based models, how about process-based models?

This thesis deeply exploited applying GANs to learn facies models from a process-based simulator, FLUMY, which is a step forward in deep generative model applications from the research to real-world challenges. This work tackled several identified problems in GAN learning 2D and 3D meandering fluvial patterns by proposing a set of unique model structures, learning frameworks and training strategies.

The ultimate product of this PhD project is a GAN-based 3D facies modelling tool for low net-to-gross meandering fluvial systems called FluvialGAN3D simulator. This project used a low NTG ratio meandering fluvial dataset as an example to develop the configuration of GANs. Extending FluvialGAN3D to other sedimentary settings requires corresponding training datasets and may need to tune GANs' hyperparameters. The FluvialGAN3D simulator consists of two pre-trained generators and a reconstruction program, achieved by solving the problems below in the thesis:

1 creating a benchmark meandering fluvial dataset available for reproduction.

2 comparison of different GAN setups.

3 generating complex multi-facies distributions that represent the features and the variability of the process-based simulations.

4 efficient GAN training on 2D patterns to reconstruct 3D facies models.

5 geological consistent 3D reconstruction of the deposited succession of arbitrary thickness.

6 investigating different extensions, including soft conditioning to well and seismic data.

Acknowledgements

Four and a half years PhD journey at Heriot-Watt University comes to an end. It is a great honour in my life to join the GeoDataScience group as a NERC CDT cohort, working with kind and knowledgeable supervisors and colleagues. I could not complete this work without the generous help and constructive suggestions from numerous people.

Firstly, I want to express my sincerest respect and appreciation to my supervision team: Prof Vasily Demyanov and Dr Daniel Arnold. Vasily, thank you for accepting me as your student, providing helpful feedback, and supporting me in numerous academic activities. No matter whether I succeeded or failed in my experiments based on the hypothesis, you always patiently helped me analyse the result and guided me to the next step. Dan, thanks for providing plentiful suggestions, not only in geology but also in scientific writing. I couldn't imagine how I completed this thesis without your strict training on polishing my writing skills.

Then, I would like to thank all NERC CDT former and current staff: Prof John Underhill, Ms Lorna Morrow, and Ms Anna Clark. This PhD project is part of the Natural Environment Research Council (NERC) Centre for Doctoral Training (CDT) in Oil & Gas [grant number NEM00578X/1], sponsored by Heriot-Watt University via their James Watt Scholarship Scheme. As the director of the CDT program, John wisely set up plenty of courses and field trips to broaden my horizon and kindly helped me in the past years. I would also like to thank Lorna and Anna, who organised fabulous courses and field trips.

I would like to thank all former and current colleagues in the GeoDataScience group and all cohorts in the NERC CDT program. Doing a PhD is creeping painfully within a gloomy tunnel, but you light my sky all the way to the bright end. Bastain, thanks for so many discussions and for sharing ideas about the academy and daily life. The Brauhaus you suggested is still my favourite pub in Edinburgh. Athos, I really enjoyed our talks and our trips together, particularly thanks for carrying me downhill when I got injured in Manchester and this thanks also to Chris, Hossein, and Abraham. Farah, thanks for all chats and for kindly sharing your knowledge with me. I am so impressed by how smart you are and your diligence. Can you believe the office is getting empty again? No worries, we knew it would recover shortly. Quentin, thanks for sharing your genius ideas in research and idiot ideas in pubs. We spent so many enjoyable trips, made so many good stories, and of course, raised so many troubles together with the rest 'idiots group' members: Denis, Luigi, Mike, Ramy, Robert, and Tom. Thank all 'idiots' for taking 'care' of me in the past years. You are the most creative group of guys I met in my PhD. I treasured all the memories of us in Aberdeen, Edinburgh, Manchester, Durham, Oxford, and Spain.

I would also like to thank all my colleagues and friends at Heriot-Watt University and other institutes who helped me at different stages of my PhD. Thank Sebastian, Florian, Andy, Helen, and Dave for your helpful discussion regarding my research. Particularly thank Patrick for your critical points on geological realism in my major review. I want to thank Ahmed, Guillaume and Zeyun for reviewing my paper and providing constructive feedback. I would also like to thank my friends, Yingfang, Qi, David, and Zaoli, for your company at different stages of my PhD. Especially, I want to thank Alan, who is a former IT supporter and incredibly efficient in helping me sort out IT issues all the time.

The rest, written in Chinese, are acknowledgements to my family.

自出国留学以来，家人亲朋无一不给予我鼓励与支持，引得多少愁眉展于笑颜，使得多少疲倦舒于戏谈。

如今得以完成学业，我最感谢的便是我的父亲孙洪波和母亲丁树习。父母不仅在经济上对我资助更在精神上促我成长。父亲常说“读万卷书，行万里路”。我便到访了欧洲多国，跟随着诸多教授考察了多地的地质。“纸上得来终觉浅”再适合不过用来形容地质建模时的心路。没有这“万里路”的支撑，纵有“万卷书”于腹也无以相互印证。母亲曾言“你喜欢的便是风景”。从此，便不再拘于他人的眼光，逍遥游于自然。生活上不对热门的事物蜂拥而至，学术上也自然不必对新颖的观点趋之若鹜。择一隅而深耕便是母亲教给我最好的一课。

同时，我想对所有关心我的亲朋好友说声谢谢。感谢祖母和外祖母对我多年来的照顾，祝您身体健康，福寿延绵。感谢两位姑姑姑父、两位姨妈姨父、两位干爸干妈、舅舅舅妈、以及诸位叔伯姑舅姨多年来对我的关心。我何其幸有几位表兄弟姐妹却胜似亲兄弟姐妹的亲人，感谢你们多年来对我以及家人的照顾。

最后，我想对天堂的爷爷问一声安好。孙儿不孝，不曾想临行前的一眼便是永远，隔屏幕的一瞥便是永别。碍于大疫未能在您床前尽孝，身后送终。愿您在天堂一切安好。

Edinburgh, March 2023

Inclusion of Published Works Form

Please note you are only required to complete this form if your thesis contains published works. If this is the case, please include this form within your thesis before submission.

Declaration

This thesis contains one or more multi-author published works. I hereby declare that the contributions of each author to these publications is as follows:

Citation details	Sun, C., V. Demyanov, and D. Arnold. "GAN learning complex fluvial facies distribution from process-based modelling." 82nd EAGE Annual Conference & Exhibition. Vol. 2021. No. 1. EAGE Publications BV, 2021.
Author 1	Conceptualization, Methodology, Software, Validation, Formal analysis, Data Curation, Writing - Original Draft, Visualization
Author 2	Supervision, Formal analysis, Writing - Review & Editing
Author 3	Supervision, Formal analysis, Writing - Review & Editing

Citation details	Sun, C., V. Demyanov, and D. Arnold. "Comparison of popular Generative Adversarial Network flavours for fluvial reservoir modelling." 82nd EAGE Annual Conference & Exhibition. Vol. 2021. No. 1. EAGE Publications BV, 2021.
Author 1	Conceptualization, Methodology, Software, Validation, Formal analysis, Data Curation, Writing - Original Draft, Visualization
Author 2	Supervision, Formal analysis, Writing - Review & Editing
Author 3	Supervision, Formal analysis, Writing - Review & Editing

Citation details	Sun, Chao, Vasily Demyanov, and Daniel Arnold. "GAN River-I: A process-based low NTG meandering reservoir model dataset for machine learning studies." Data in Brief 46 (2023): 108785.
Author 1	Conceptualization, Methodology, Software, Validation, Formal analysis, Data Curation, Writing - Original Draft, Visualization
Author 2	Supervision, Formal analysis, Writing - Review & Editing
Author 3	Supervision, Formal analysis, Writing - Review & Editing

Citation details	Sun, Chao, Vasily Demyanov, and Daniel Arnold. "Geological realism in Fluvial facies modelling with GAN under variable depositional conditions." Computational Geosciences 27.2 (2023): 203-221.
Author 1	Conceptualization, Methodology, Software, Validation, Formal analysis, Data Curation, Writing - Original Draft, Visualization
Author 2	Supervision, Formal analysis, Writing - Review & Editing
Author 3	Supervision, Formal analysis, Writing - Review & Editing

Contents

List of Publications

Sun, C., Demyanov, V., & Arnold, D. (2023).A Conditional GAN-based Approach to Build 3D Facies Models Sequentially Upwards. Computers & Geosciences, In Review.

Sun, C., Demyanov, V., & Arnold, D. (2023). Geological realism in Fluvial facies modelling with GAN under variable depositional conditions. Computational Geosciences, 1-19.

Sun, C., Demyanov, V., & Arnold, D. (2023). GAN River-I: A process-based low NTG meandering reservoir model dataset for machine learning studies. Data in Brief, 46, 108785.

Sun, C., Demyanov, V., & Arnold, D. (2021, October). Comparison of popular Generative Adversarial Network flavours for fluvial reservoir modelling. In 82nd EAGE Annual Conference & Exhibition (Vol. 2021, No. 1, pp. 1-5). European Association of Geoscientists & Engineers.

Sun, C., Demyanov, V., & Arnold, D. (2021, October). GAN learning complex fluvial facies distribution from process-based modelling. In 82nd EAGE Annual Conference & Exhibition (Vol. 2021, No. 1, pp. 1-5). European Association of Geoscientists & Engineers.

Chapter 1

Introduction

1.1 Background

Complex sedimentary systems, such as fluvial systems, introduce reservoir scale heterogeneity over hundreds of meters, impacting the flow response. The distributions of rock physical properties, like porosity and permeability, control the reservoir dynamics, fluid in place and flow [Corbett, 2012]. Fluvial reservoirs, particularly low net to gross fluvial reservoirs, can have variably connected sand-bodies vertically and laterally, resulting in variable flow paths, impacting the areal and vertical sweep [Corbett, 2012].

In practice, the modellers often use a facies model to honour the reservoir scale heterogeneity [Walter, 1984]. A facies model is a common tool to summarise the sedimentary environment, where each facies refer to a specific body with consistent characteristics reflecting a certain sedimentary condition, process or environment [Miall and Miall, 2016, Reading, 1978, Walker, 1976]. Fluvial systems, such as meandering systems, create multiple facies with different geometries and spatial distributions that result from geological processes, like channel migration. Facies transition, change from one facies to another, along the channel centreline, can add geological complexity in terms of geological bodies, particularly in a low NTG and moderate aggradation rate environment, e.g., point bar sand usually only accretes at the inner bank of the channel. Meandering systems feature different channel fill and boundary facies, complicated facies transition, non-constant facies geometry, fluctuant channel sinuosity, and variable grain size distributions. These complex geometrical heterogeneities, e.g. curvilinear, are challenging to model with conventional and even advanced statistical algorithms. Therefore, efficiently developing a geologically plausible facies model with a high-level complexity conditioned to data is often challenging.

1.1.1 Fluvial Systems

Fluvial systems result from various upstream and downstream controls, including autogenic constraints (e.g. overbank flooding, avulsion, channel migration, etc.) and allogenic constraints (e.g. tectonics, climate, sediment type, valley gradient, base level, etc.) [Ethridge and Schumm, 2007, Schumm, 2007]. The joint effect of those elements yields different morphology. One classification categories channels into four types: straight, meandering, braided and anastomosing [Miall, 1977]. Different types of channels present contrasting patterns in sinuosity, the number of channels, etc. (see Figure 1.1). Besides the channel morphology in plane view, those autogenic and allogenic factors of fluvial systems also significantly impact the sand bar distribution that dominates the reservoir heterogeneity. For example, the meandering river accretes sand bars (point bars) at the inner bank, while the braided rivers form sand bars in the middle of the channel (braid bars) and are generally wider and shallower than meandering rivers when they have a similar discharge as the braided rivers are more bed-load dominated rivers [Ferguson, 1984]. Therefore, the modellers need to involve necessary geology to reflect realism when modelling fluvial systems.

Geological realism often refers to the specific geological knowledge considered for a subsurface investigation [Wellmann and Caumon, 2018]. Complex processes yield the spatial distribution of properties over long time periods, making it hard to consider all aspects of the processes when modelling the distribution. Thus, the modellers must decide how much geological realism is needed for a geomodel. For facies modelling, modellers often consider the geometry, arrangement of deposited facies, facies proportion and facies connectivity as the principal aspects of geological realism to be preserved [Wellmann and Caumon, 2018]. When the sedimentary system is relatively simple, more likely to be homogenous or has a clear trend, the modellers might use a simple correlation model to

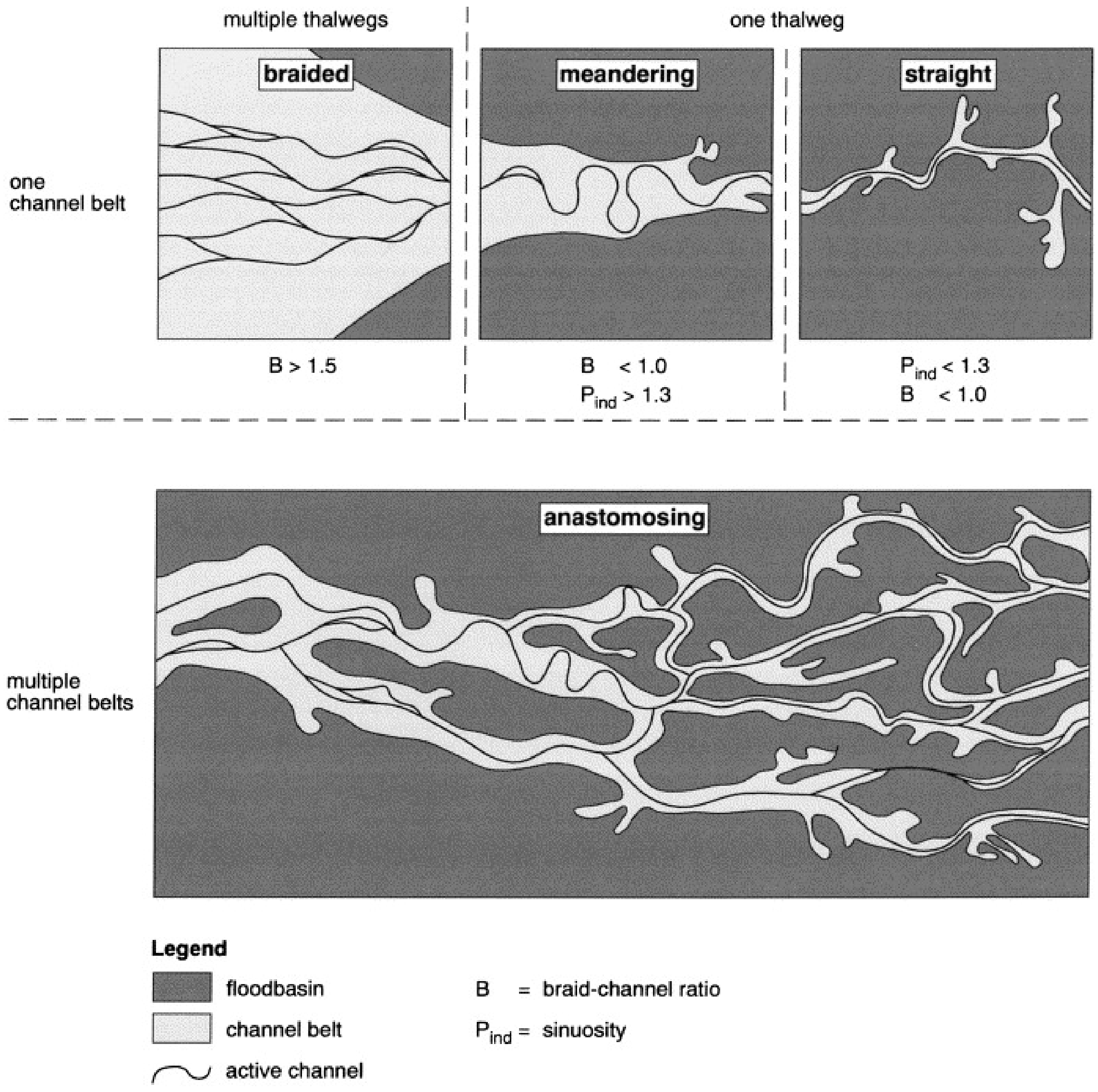


FIGURE 1.1: Different types of fluvial patterns. From Makaske [2001]

represent it. As for fluvial systems that are highly heterogenous, the modellers need to consider more complicated tools to model them.

1.1.2 Facies Modelling Tools

Conventional facies modelling tools include process-based models and geostatistical approaches. In general, a broad classification categorises geostatistical approaches into two types: pixel-based and object-based. The pixel-based method, for example, two-point statistics (Sequential Indicator Simulation, SIS, TGS, PGS) or Multi-point Statistics (MPS), creates realisations by assigning values pixel by pixel [Deutsch et al., 1992, Strebelle, 2000]. In contrast, the object-based model needs to define the object shape in advance and place the object randomly on the modelled ground [Maharaja, 2008]. Alternatively, physics-driven process-based models simulate realisations following the physics equations/rules/simplified processes of the depositional process derived from natural geology observations or laboratory experiments [Bogaart et al., 2003, Grimaud et al., 2022, Lopez, 2003].

However, the approaches above either lose realism due to inaccurate estimation of facies' spatial distribution or are slow or inflexible to data conditioning. SIS/TGS/PGS and MPS are limited in how well they can reproduce geometries to the high level of multi-facies complexity without losing realism as their spatial correlation models represent simplified facies distributions, though MPS can capture more complex spatial correlation than SIS/TGS/PGS. The object-based model, e.g. TiGenerator [Maharaja, 2008], can only simulate pre-defined shapes while ignoring the impact of sedimentary processes and the wide variation of the facies geometries in nature [Cojan et al., 2005, Zhang et al., 2019b]. The process-based models can simulate multiple plausible depositional scenarios but are very computationally costly and are difficult to condition to the data point and seismic [Bogaart et al., 2003, Bubnova, 2018, Strebelle, 2021]. Process-based approach, therefore, suits to elicit the uncertainties from the system parameters, which can then provide the geometric priors for either object-based modelling or MPS if the number of facies reduced to a small amount, e.g. three or four [Strebelle, 2021]

Recent publications demonstrated that deep generative models, e.g. Generative Adversarial Networks (GANs) and Variational Auto-Encoder (VAE), could efficiently parameterise and reproduce sinuous channel geometries and associated linear facies transitions in a shale background conditioned to multi-types observed data [Azevedo et al., 2020, Chan and Elsheikh, 2019, Laloy et al., 2017, 2018, Song et al., 2021a,b, Zhang et al., 2019b]. Those VAE and GAN applications either used Object-Based Models or MPS as the training dataset, yielding their deep generative models could only reproduce data at the same level of complexity as their training dataset, e.g. geometries of parametric objects [Azevedo et al., 2020, Laloy et al., 2017, 2018]. Most deep generative workflows do not hard condition their simulations to measured data such as wells facies log [Chan and Elsheikh, 2019, Zhang et al., 2021, 2019b]. Simulating facies realisations that fit precisely observed data is a feature of geostatistical modelling to build realisations of the inter-well uncertainty and, therefore, should be an advanced feature of other reservoir facies modelling approaches. Indeed, the observed data in the wells may not represent the upscaled grid cell property when the grid dimensions are hundreds of meters. Thus, if a facies model should precisely honour the observed data is controversial, but generally, data conditioning should be an option in reservoir facies modelling tools in case of strong confidence in certain geogrids' value.

1.2 Objectives

Encouraged by earlier successes in deep generative model applications to modelling channelised reservoirs, this PhD project further applies GANs to model more complex fluvial systems than earlier GAN applications [Chan and Elsheikh, 2019, Laloy et al., 2018, Zhang et al., 2019b]. Compared to the training dataset used in earlier applications [Chan and Elsheikh, 2019, Laloy et al., 2018, Zhang et al., 2019b], this study adds more facies (e.g. point bar and different channel fills) to the training dataset that introduces more complicated facies transitions and a bigger variation of the facies' geometry. To investigate how many plausible geological features GANs can learn from a process-based model's realisations produced by different avulsion settings, this thesis uses meandering fluvial systems as an example to test and further develop GANs to produce more complex field-scale realisations. To extend the GAN application to 3D simulation with a more generalised usage, a further study researches potential ways to reuse pre-trained models, avoiding wasting time preparing datasets and training GANs after moving to a field with similar sedimentary environments but different field sizes.

1.2.1 General Objective

The main objective of this study is to investigate how to apply GAN to learn complex 3D meandering facies models from process-based models.

1.2.2 Specific Objectives

To achieve this general objective, developing a series of new tools is necessary to fulfil the following specific objectives:

- Create a training dataset with a state-of-art process-based algorithm to represent uncertainty across a range of plausible meandering depositional scenarios.
- Train a GAN-based simulator to reproduce plausible 2D realisations covering a wide range of model diversity by selecting an appropriate GAN baseline via a comparison study and further developing it to handle identified challenges associated with the training dataset.
- Recreate the complex geology in 3D effectively with a handy control on the reservoir thickness.
- Explore existing data conditioning techniques to blend conditioning data into GAN generations.

1.3 Outlines

The rest of this thesis follows the organisation below:

- Chapter 2 reviews three popular classes of facies modelling tools, including process-based approaches, geostatistical approaches and machine learning-based approaches. This chapter recaps the main mechanism for every simulation tool, shows typical simulation examples from previous publications, and discusses the advantages and disadvantages. Particularly, this chapter provides a detailed review of artificial intelligence components in popular deep generative models, explaining the terms and algorithms used in later chapters.
- Chapter 3 presents the benchmark datasets created by a process-based model. A process-based algorithm, FLUMY [Grimaud et al., 2022], simulates low net-to-gross meandering models with different avulsion rates, composing a sterner training dataset for GANs named GAN River-I. This dataset serves as the benchmark for the studies in the rest chapters, which discuss investigations of GAN learning in 2D and 3D facies modelling cases. Several qualitative and quantitative analyses describe GAN River-I and introduce tools for measuring geomodels, which are used for evaluating GAN performance in later chapters.

- Chapter 4 introduces the research on developing Fluvial GAN, a GAN variant designed for 2D meandering fluvial facies modelling. This chapter starts with comparing popular GAN variants in learning 2D 3-facies meandering patterns and then applies the best GAN candidate to multi-facies modelling. This chapter identifies two typical unrealistic features in GAN generations and further develops three improvements to help GAN better learn multi-facies meandering models.

- Chapter 5 describes the 3D extension of Fluvial GAN with a flexible output facies model thickness based on a conditional normalisation technique. This chapter presents a novel 3D reconstruction process to simulate the facies model slice by slice upward vertically, conditioning the upper layer to its lower layers weighted by a decaying mechanism. Two proposed training enhancements improve the geological realism and diversity in reconstructed models. This chapter also discusses a potential way of varying the 3D meandering fluvial patterns with an external parameter, making the resulting difference more explainable.

- Chapter 6 investigates how to bridge the gaps between research staged Fluvial GAN to real field applications, including latent (parameterised) size and data conditioning. This chapter discusses the impact of latent size on Fluvial GAN's performance by reducing its original latent size (default value in Fluvial GAN) to a small value that is manageable in many heuristic-based global optimisers. Another study explores several data conditioning techniques for enforcing GANs' generation to honour observed data, showing successes and challenges.

- Chapter 7 remarks on the conclusions of this PhD project and suggests future directions in developing GAN-based facies simulation. This chapter briefly summarises the main findings and identified challenges during the PhD study. Depending on the purpose of creating facies models, GAN-based facies simulation potentially have diverse routes of improvement, which are discussed at the end of this thesis.

Chapter 2

Facies Modelling Approaches for Meandering Systems

Geostatistical simulations, e.g.[Remy et al., 2009] and process-based models, e.g.[Lopez, 2003] are two classical tools for modelling fluvial systems, and machine learning-based algorithms (deep generative models) proved their capability of learning complex geological patterns in recent years [Chan and Elsheikh, 2019, Laloy et al., 2018, Zhang et al., 2019b]. Due to geological pattern complexity, fluvial systems, such as meandering systems, present substantial heterogeneity that needs a proper facies model to represent. Because the number, proportion, spatial distribution and type of reservoir facies present typically control the distribution of porosity, permeability, and flow behaviour. Depending on the purpose, modellers choose different tools to model the facies distribution, e.g., process-based model, object-based model, multi-point statistics, etc.

This chapter will briefly overview a wide range of algorithms used to model fluvial facies to date, including their algorithms, advantages and disadvantages. This chapter will also review the particular sedimentary facies features that are essential to reproduce by geostatistical or generative learning model. This thesis will focus on reproducing these key spatial features with the proposed Fluvial GAN in the following chapters.

2.1 Process-based Model

Process-based approaches represent geological sedimentation processes with mathematical models derived from observations of nature and laboratory experiments [Lopez, 2003]. The river meanders in fluvial systems result from channel evolution with erodible banks, which raises challenges in modelling facies distribution due to the complex river evolution mechanism. A set of empirical or theoretical formulas that describe sedimentary processes of forming meandering systems composes the numerical simulation of the meandering of the river. Depending on the model complexity, process-based models for meandering rivers may incorporate various processes relevant to climate, hydrology, soil erosion, channel geometry and sediment transport [Bogaart et al., 2003]. Meanwhile, channel migration and sediment deposition are essential for producing 3D fluvial facies models.

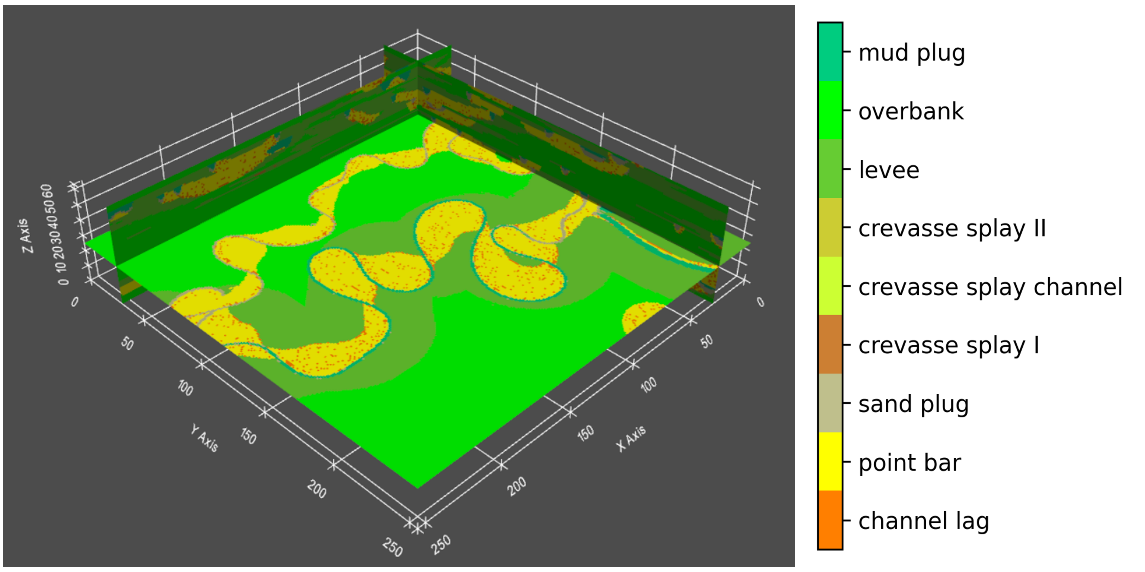


FIGURE 2.1: An example of facies model from $FLUMY^{TM}$ [Grimaud et al., 2022], a process-based model.

Process-based approaches can create useful conceptual geological models but are usually computationally costly and challenging to condition to data. Process-based models simulate sedimentary processes to create associated facies at each time step, reflecting the realism of considered physics and processes. Therefore, the time cost of building a sequence depends on the calculation speed at each time step and the sedimentation rate that determines how many steps it takes to build up a sequence with the pre-defined thickness. Simulated sequence results from the deposited sedimentary facies during the simulation period, while observations from well and seismic are spatial, not temporal. This discrepancy makes data conditioning, especially for both point conditioning to well and soft conditioning to seismic, challenging as the models need to consider the process realism at the current time step and observed data at the location simultaneously, which sacrifices both geological realism and conditioning accuracy [Bubnova, 2018].

2.1.1 Channel Migration and Bend Theory

Channel migration prediction relies on hydrodynamics and sediment transport hypotheses, the St. Venant equations [de Saint-Venant et al., 1871]. Ikeda et al. [1981] invented the bend theory that linearly approximates the lateral bend amplitude increase based on near bank velocity perturbation (see Figure 2.2). Later improvements in bend theory make computer simulations successfully infer the long-term behaviour of meandering rivers based on channel centerlines [Johannesson and Parker, 1989, Lopez, 2003, Parker et al., 2011, Sun et al., 2001].

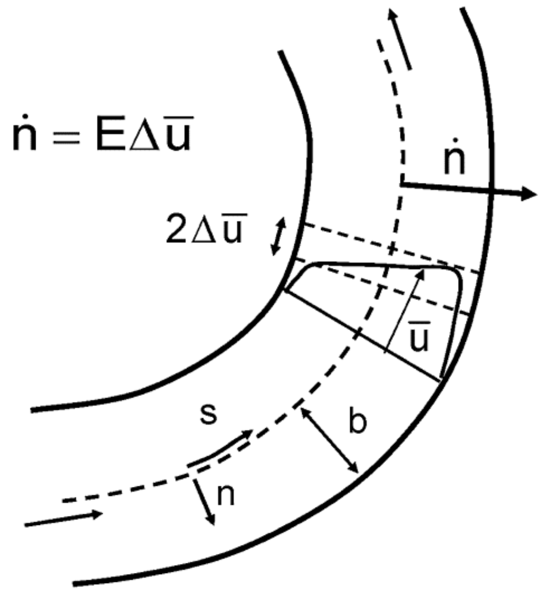


FIGURE 2.2: A schematic diagram of bend theory. From Bubnova [2018], Parker et al. [2011].

In bend theory, Ikeda et al. [1981] transformed the channel centerline to Cartesian coordinates and calculated velocity perturbation at each point, which yielded the lateral migration of river meanders. Ikeda et al. [1981] estimated near bank velocity perturbation using equation 2.1.

$$U\frac{\partial u_b'}{\partial \tilde{s}} + 2\frac{U}{H}C_f u_b' = b[-U^2\frac{\partial \varphi'}{\partial \tilde{s}} + C_f\varphi'(\frac{U^4}{gH^2} + A\frac{U^2}{H})] \tag{2.1}$$

where u_b' is near bank velocity perturbation, $\tilde{s}$ is coordinates along the channel centerline, U is reach-averaged tangential velocity, H is reach-averaged depth, C_f is fraction factor, φ' is centerline curvature perturbation, b is normal half-width, g is acceleration of gravity, and A is an $O(1)$ factor. The lateral migration rate is a product of velocity perturbation and bank erosion coefficient. By linearising, the bend theory estimates the channel migration at the next time step based on the current channel pattern and flow velocity.

2.1.2 Aggradation

Aggradation refers to sediment deposition during periods of high discharge due to the lower velocities, which raises the base level and shows a big spatial difference across the field [Nicholas and Walling, 1996]. A sediment supply greater than the transport capacity triggers aggradation, which drops large particles first and then fine deposits. During floods, suspended load covers the lowland area known as the floodplain and deposits more rapidly nearby channels forming an elevation called natural levee (see Figure 2.3), higher than distal floodplain [Saucier, 1994]. The aggradation rate decreases from channels to farther floodplains and differs between flood events [Ferring, 1986, Nicholas and Walling, 1996].

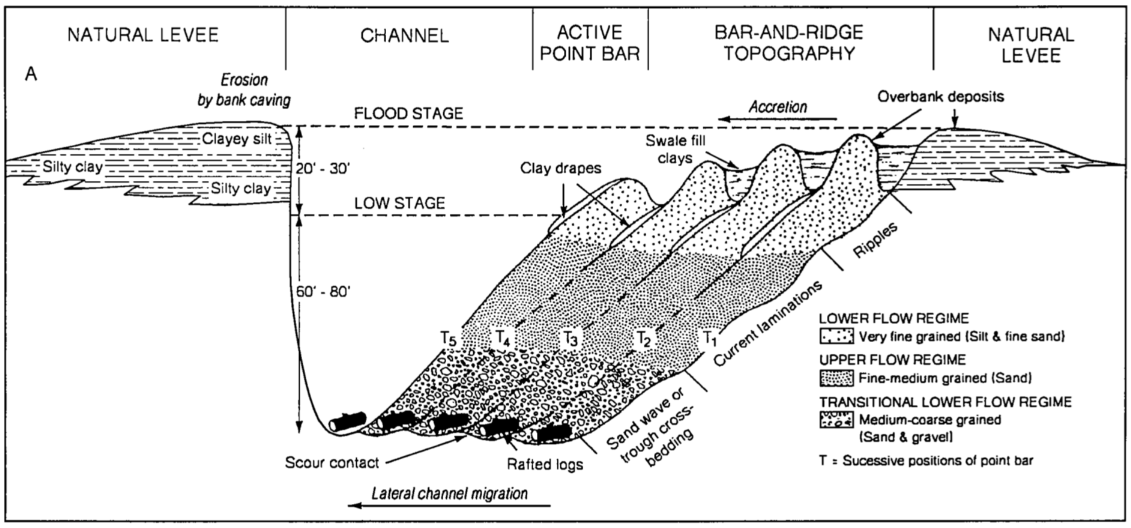


FIGURE 2.3: A cross-section of a meander belt. From Saucier [1994].

2.1.3 Levee breach, Avulsion and Meander Cutoff

Levee breach leads to water and sediment in the floodplain, triggering crevasse splay deposition and may cause some forms of channel avulsion [Nienhuis et al., 2018]. Figure 2.4 shows two possible results of natural levee breaches in the Mississippi River. Once a levee breach occurs, suspended sediment flows outside the channel and levee. Sediment forms fan-shaped crevasse splay deposition if the breach healed. Otherwise, the breach may induce a channel avulsion.

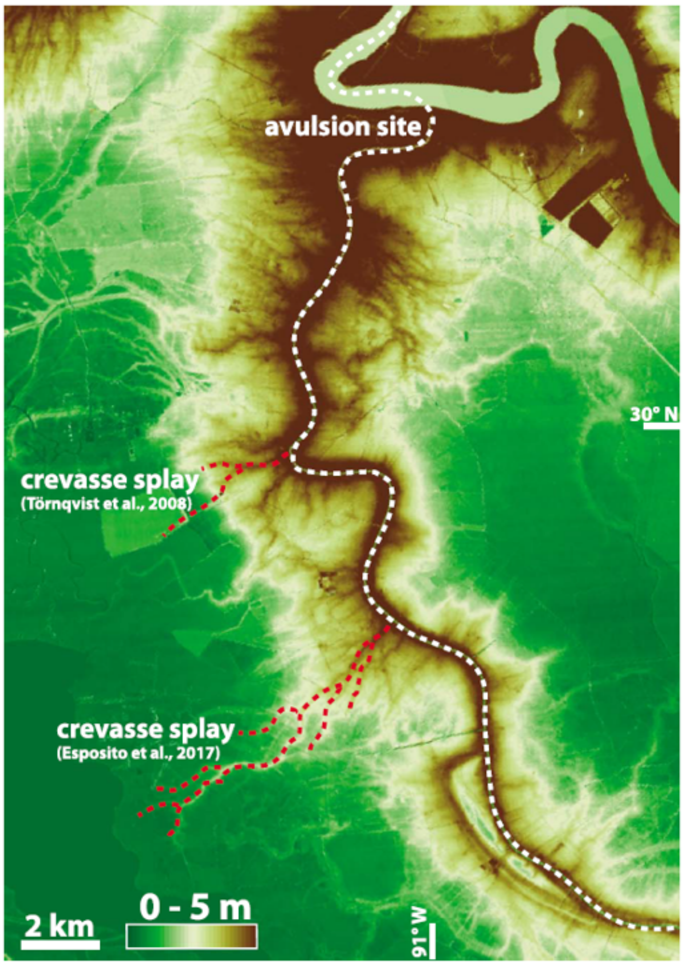


FIGURE 2.4: LIDAR image of crevasse splay and avulsion caused by natural levee breaches. From Nienhuis et al. [2018].

Avulsion is the process during which a meandering river abandons its old channel for a new one [Heller and Paola, 1996]. Slingerland and Smith [2004] summarised three styles of avulsions, including annexational avulsion, incisional avulsion and progradational avulsion, and categorised avulsions as full versus partial, nodal versus random and local versus regional according to the behaviours of the diverted flow (see Figure 2.5). The whole flow changes the channel leading to full avulsion, while a portion of the flow runs into the new channel known as partial avulsion [Slingerland and Smith, 2004]. Depending on the avulsion site, nodal avulsions abandon channels around the fixed area, but random avulsions may abandon channels anywhere along the active channel [Slingerland and Smith, 2004]. The main difference between local and regional avulsions is whether the new channel rejoins the original channel downstream [Heller and Paola, 1996].

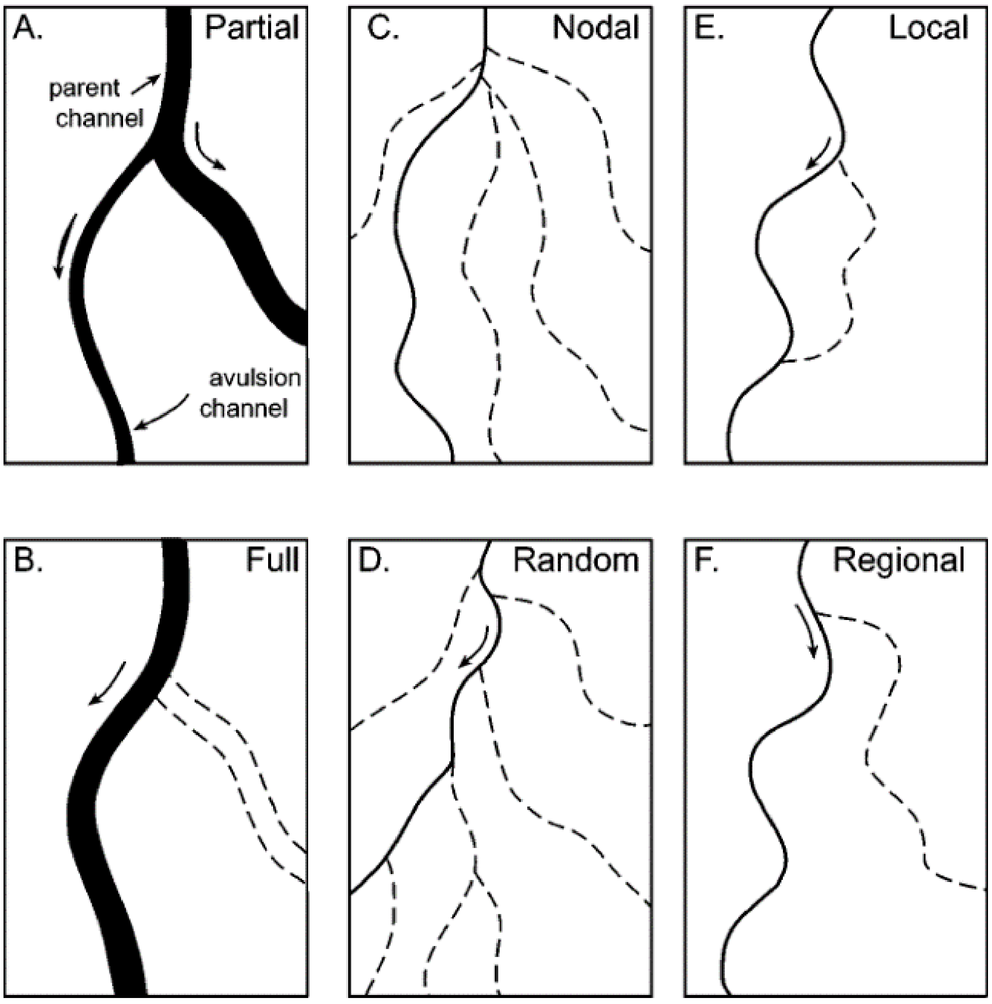


Figure 2.5: Illustration of different categories of avulsion. From Slingerland and Smith [2004].

Meander cutoff is another type of channel abandonment that forms a new channel by connecting two sides of a meander loop and abandoning the old channel, impacting point-bar geometries and heterogeneity [Russell, 2017]. Neck cutoff and chute cutoff are two common meander cutoffs. When a meander overdevelops, the upstream and downstream limbs move close and converge to form a typically high diversion angle neck cutoff benefiting more mud-prone channel-fill deposits [Russell, 2017]. In contrast, chute cutoff builds on the point bar or old abandoned channels, which often results in a low diversion angle and keeps channels active for a longer time favouring sand-prone channel-fill deposits [Russell, 2017]. Russell [2017] summarised the meander cutoffs based on their abandonment mechanism and post-cutoff meander patterns (see Figure 2.6).

2.2 Geostatistical Approach

Geostatistical methods simulate stochastic realisations estimating the reservoir property away from well locations under some spatial correction assumptions. By using some conceptual model to describe the spatial correlation of the facies, pixel-based geostatistical approaches simulate stochastic realisations of the facies' spatial distribution sequentially through the modelled domain, i.e., two-point statistics (Sequential Indicator Simulation, SIS, TGS, PGS) or Multi-point Statistics (MPS) [Deutsch et al., 1992, Remy et al., 2009, Strebelle, 2000]. Another popular geostatistical approach is the object-based model, which creates realisations via dropping pre-defined objects into the simulated domain and adjusting them to conditioning data [Maharaja, 2008, Remy et al., 2009]. This section only reviews the classical version of the three geostatistical methods instead of diving deeper into more advanced versions.

2.2.1 Two-point Statistics

Two-point statistical methods are based on a variogram model that describes the spatial correlation under the secondary stationary assumption [Deutsch et al., 1992]. A traditional measure of the variogram, rigorously known as semivariogram, calculates half of the mean squared difference between the values separated by a lag distance, given by Equation 2.2 [Deutsch et al., 1992]

$$\gamma(h) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} (x_i - y_i)^2 \tag{2.2}$$

where h is the lag distance, N is the number of point pairs, i denotes the index of pairs, x and y are the start and end points, respectively. After computing the variogram along a direction, a user-defined analytical, functional model, e.g. the spherical, exponential, and Gaussian model, is fitted to the calculated variogram for producing the variogram model.

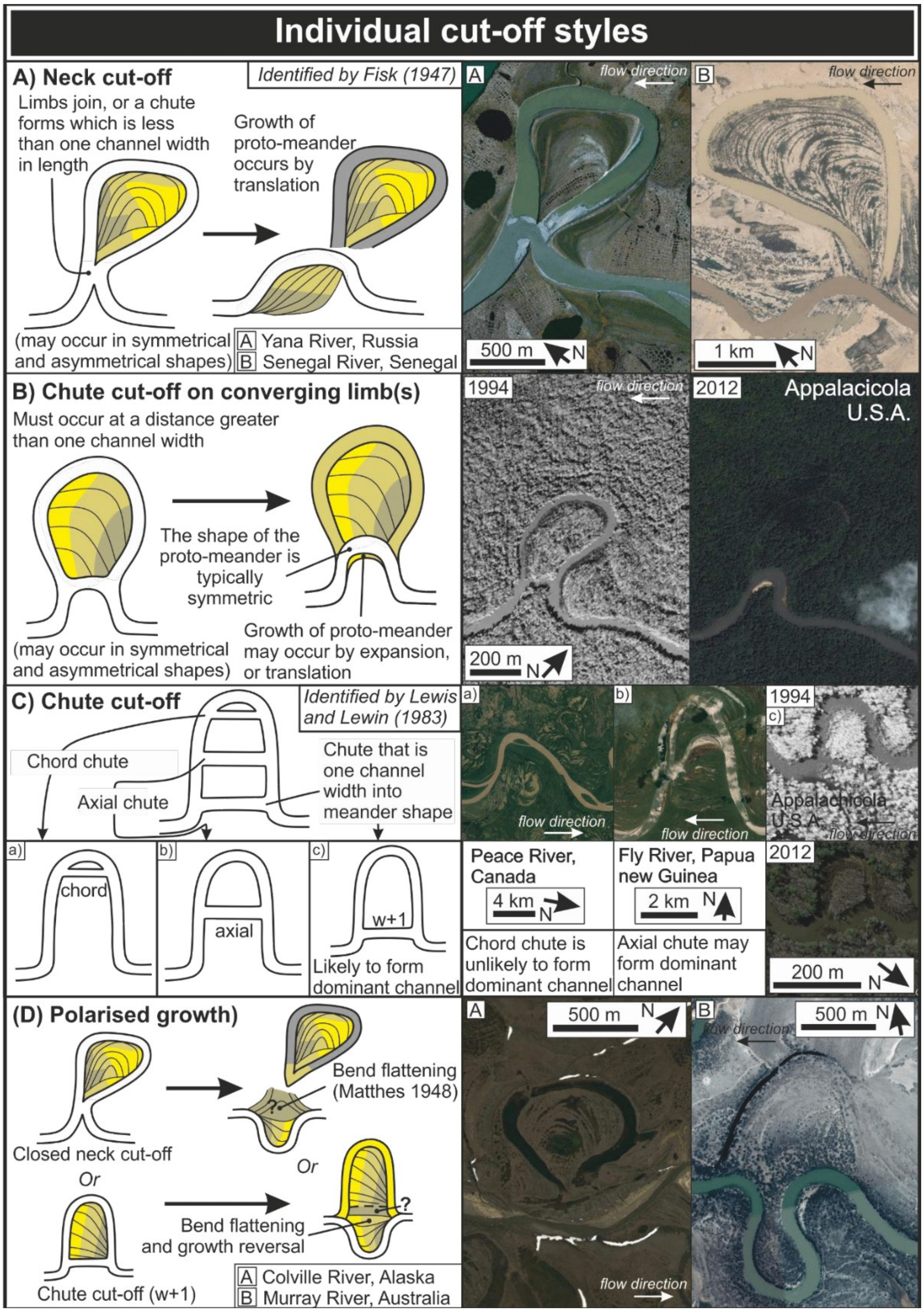

Individual cut-off styles
A) Neck cut-off
Identified by Fisk (1947)
Limbs join, or a chute forms which is less than one channel width in length
Growth of proto-meander occurs by translation
(may occur in symmetrical and asymmetrical shapes)
A Yana River, Russia
B Senegal River, Senegal
flow direction
500 m N
1 km N
B) Chute cut-off on converging limb(s)
Must occur at a distance greater than one channel width
The shape of the proto-meander is typically symmetric
(may occur in symmetrical and asymmetrical shapes)
Growth of proto-meander may occur by expansion, or translation
1994
2012
Appalacicola U.S.A.
200 m N
C) Chute cut-off
Identified by Lewis and Lewin (1983)
Chord chute
Axial chute
Chute that is one channel width into meander shape
chord
axial
w+1
Likely to form dominant channel
Peace River, Canada
4 km N
Chord chute is unlikely to form dominant channel
Fly River, Papua new Guinea
2 km N
Axial chute may form dominant channel
Appalachicola U.S.A.
200 m N
(D) Polarised growth)
Closed neck cut-off
Or
Bend flattening (Matthes 1948)
Bend flattening and growth reversal
Chute cut-off (w+1)
A Colville River, Alaska
B Murray River, Australia
500 m N

FIGURE 2.6: Different types of meander cutoffs. From Russell [2017].

The variogram model determines the spatial continuity of the properties along a chosen direction in the field. Then, a stochastic sequential simulation populates a value for every cell based on the variogram models using a linear interpolation estimator, kriging (see Figure 2.7).

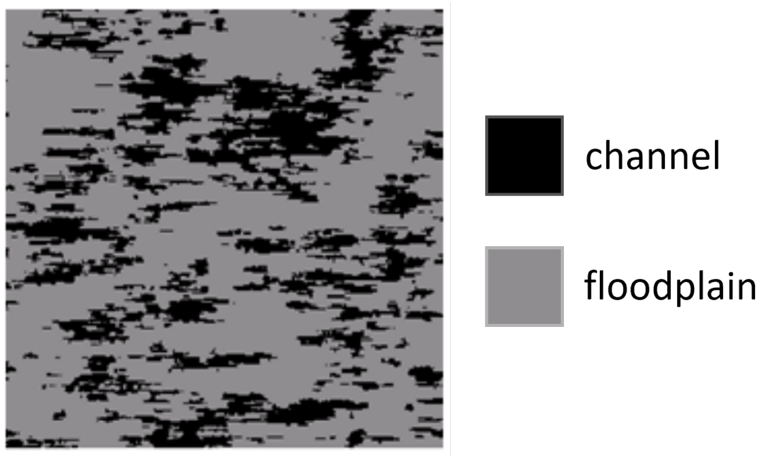


FIGURE 2.7: An example of a realisation from sequential indicator simulation (SIS), a popular two-point statistics method used for producing facies models. Modified after Remy et al. [2009].

Regarding facies modelling, variogram-based approaches need to use probability values to code the facies variables in the indicator format [Pyrcz and Deutsch, 2014]. This conversion simplifies the data integration when the program processes categorical data, e.g. facies variable. Common variogram-based methods for facies modelling include Sequential Indicator Simulation (SIS), Truncated Gaussian Simulation (TGS), Puri-Gaussian Simulation (PGS).

Sequential indicator simulation (SIS) visits every cell in a random path, where it draws a value from the probabilities estimated by a kriging estimator, a linear interpretation based on the variogram model [Pyrcz and Deutsch, 2014, Remy et al., 2009]. SIS calculates the variogram using both well data and simulated cells in the previous step. As the indicator method calculates the probability of converting from one facies to the other while not including the order relationships, SIS may assign a facies at any place and lack the facies sequence control of condition its realisation to the depositional facies boundary, resulting in noisy realisation (see Figure 2.8 a) [Pyrcz and Deutsch, 2014, Yarus et al., 2012].

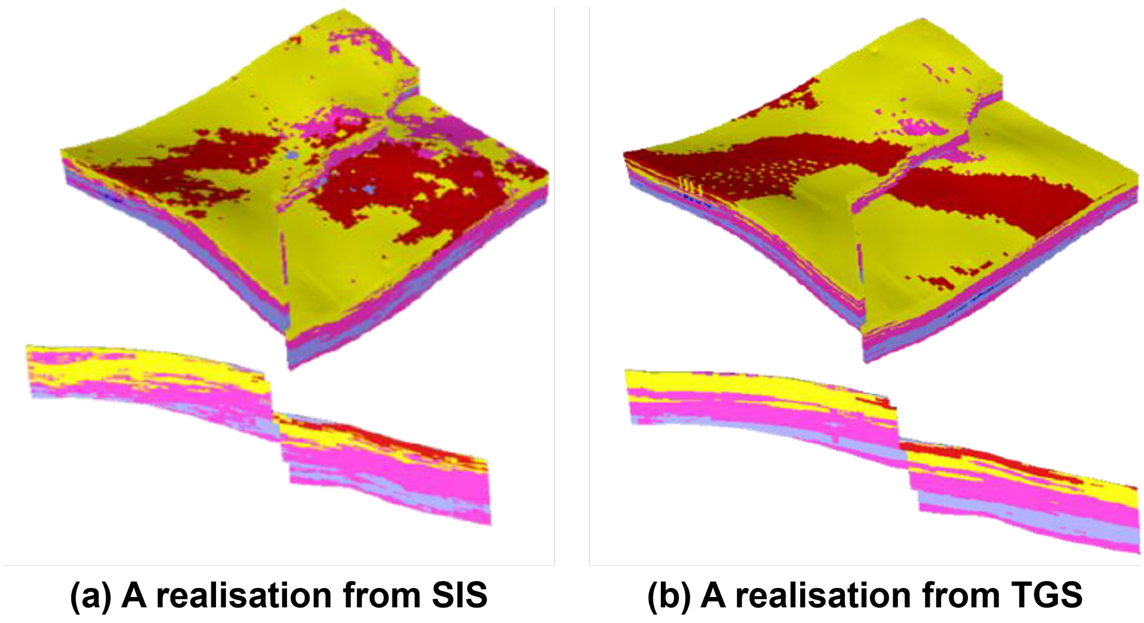


FIGURE 2.8: Comparison of realisations from SIS and TGS. Modified after Yarus et al. [2012].

Truncated Gaussian simulation (TGS), on the other hand, is another variogram-based approach for facies modelling that is a parametric algorithm based on the Gaussian distribution of facies attribute and enables incorporating facies order information. TGS sets threshold values as the boundary between facies at a Gaussian distribution to convert a realisation from a continuous Gaussian variable to categorical facies [Pyrcz and Deutsch, 2014, Remy et al., 2009]. This algorithm works as a post-processing of variogram-based Gaussian simulation, which allows the user to arrange the facies in order and vary the threshold locally. Compared to SIS, TGS performs better in dealing with ordinary categorical facies (see Figure 2.8 a and b), whose realisations have a clearer boundary between facies.

Puri-Gaussian simulation (PGS) is a more advanced version of truncated Gaussian simulation, which extends TGS to multi-Gaussian fields to represent a more comprehensive facies transition. PGS also needs to set the conversion rules between the continuous Gaussian field and categorical facies, but it can use more than one variogram to describe the spatial correlations between different facies [Mariethoz et al., 2009, Pyrcz and Deutsch, 2014, Yarus et al., 2012]. Figure 2.9 shows an example of PGS realisation.

Variogram-based methods, in general, can reproduce spatial continuity and the fraction of

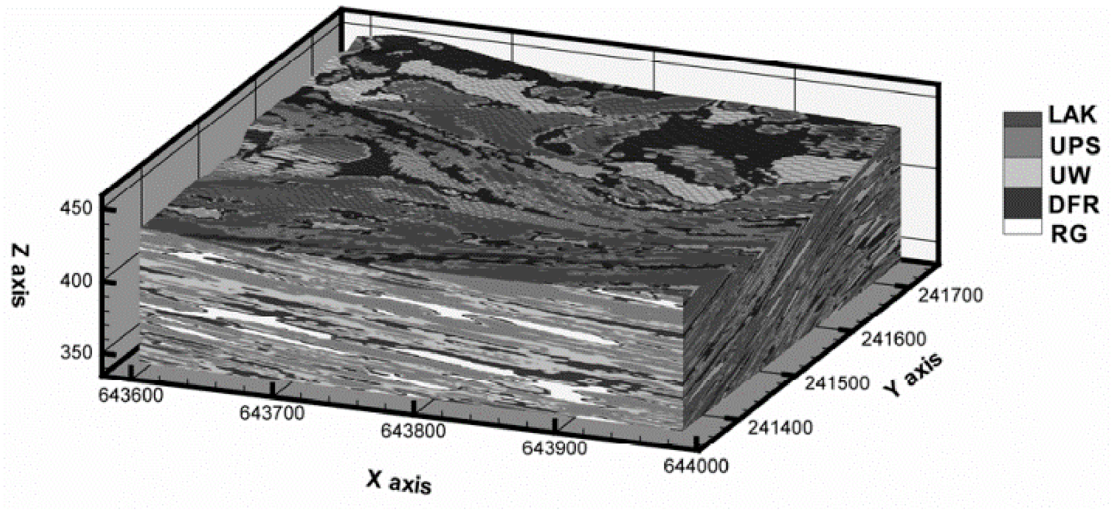


FIGURE 2.9: An example of realisations from PGS. Modified after Mariethoz et al. [2009].

facies and are suitable for conditioning diverse types of data as they generate realizations one pixel at a time. However, the variogram represents relatively simple spatial correlations. Therefore, two-point statistics have difficulty reproducing geometries with complex connectivity patterns, such as meandering channels, which may cause the simulated realizations to fail to capture the real connectivity.

2.2.2 Object-based Model

Object-based methods simulate facies distribution by dropping the pre-defined geobodies, such as channels, levees and lobes, one by one onto simulation grids [Deutsch and Wang, 1996]. Object-based methods can create plausible geological geometry of facies based on the prior geological description (see Figure 2.10).

However, a couple of drawbacks limit its application. One weakness of object-based methods is that they are difficult to data conditioning to the combination of point and soft conditioning probability map data. Although the modellers can condition realisations from object-based methods to well data using Markov Chain Monte Carlo algorithms, the convergence is slow and may fail to converge [Hauge et al., 2007]. Another issue is that the modellers must develop a different algorithm for each new object type [Strebelle et al., 2003], which is limited to objects whose shapes can be parameterized [Zhang et al., 2019b]. Also, objects are merely simplifications of the real geo-bodies, the exact same

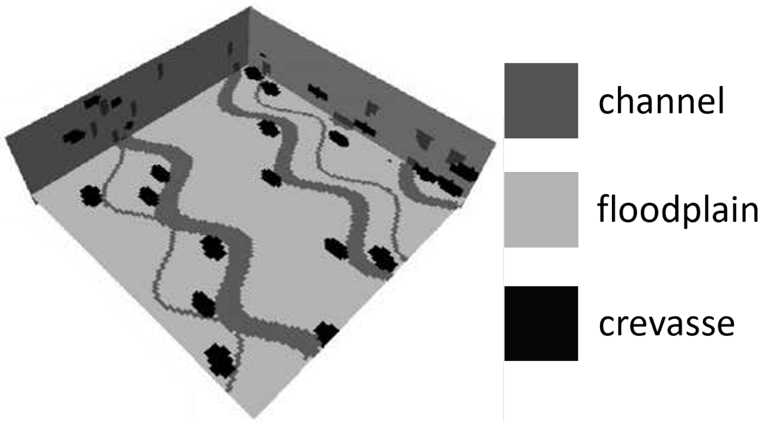


FIGURE 2.10: An example of a facies model from TiGenerator, an Object-based model. Modified after Maharaja [2008].

thing with process-based approaches like FLUMY. The variety and combinations of geobodies' shapes in nature are far more than an algorithm can represent in detail, even with a certain amount of stochasticity.

2.2.3 Multi-point Statistics

Multi-point statistics (MPS) is a class of geostatistical approaches that sequentially simulate realisations by drawing values based on the conditional probabilities calculated from a training image (TI) [Pyrcz and Deutsch, 2014]. Though their ways of deriving, storing, and restituting patterns are different, MPS algorithms generally contain basic steps, including getting patterns from the TI, defining a simulation path and the node to be simulated, finding the node's neighbourhoods, calculating the conditional distribution of the variable at current node conditioned to its neighbourhoods, and sampling a value from this conditional distribution [Mariethoz and Caers, 2014].

MPS algorithms have different implementations to learn from the TI to restitute simulations. Guardiano and Srivastava [1993] first came up with the concept of MPS simulation, which learned the spatial correlations from training image (TI) instead of the variogram model. This MPS algorithm is computationally costly because it needs to scan the whole TI for every new simulated node [Guardiano and Srivastava, 1993, Mariethoz and Caers, 2014, Pyrcz and Deutsch, 2014]. The first practical implementation of MPS is the single normal equation simulation (SNESIM) by Strebelle [2000], who improved the original

MPS by using search trees to store pre-computed conditional proportions before simulation. SNESIM defines a template that is commonly an ellipsoidal shape with orientation and a number of points around a central node to calculate the conditional probability of the variable value at the central node. The location and value of those nodes within a template compose a data event, which is what SNESIM and many other MPS algorithms learn from the TI. SNESIM stores the learned information in a search tree where it retrieves the conditional probabilities to simulate realisations [Mariethoz and Caers, 2014, Pyrcz and Deutsch, 2014, Strebelle, 2000]. From this learning perspective, MPS algorithms learn a model from data and store knowledge in a tree, list or other forms of a dictionary, which is consistent with the machine learning-based approach.

Compared to variogram-based methods, MPS can capture multi-point-based structural information and therefore simulate more complex geological patterns, for example, channels (see Figure 2.11). As MPS also simulate realisation pixel by pixel or patch by patch, it can blend multiple conditioning data into its simulation [Liu et al., 2005].

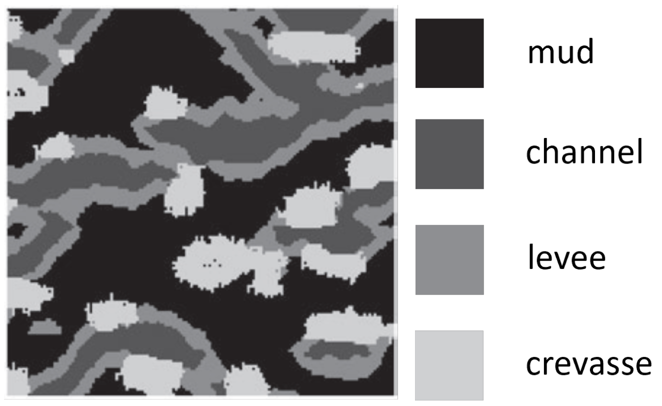


FIGURE 2.11: An example of a facies model from SNESIM, an MPS approach. Modified after Remy et al. [2009].

However, MPS algorithms often fail to reproduce realistic non-linear patterns, especially for non-stationary simulations conditioned to well data [Zhang et al., 2019b]; see Figure 2.12. Chan and Elsheikh [2019] also mentioned that MPS doesn't provide a convenient parameterisation of its simulated realisations, which is an issue of many geostatistical approaches but essential for many existing pipelines, for example, optimisation approaches in the history matching loop.

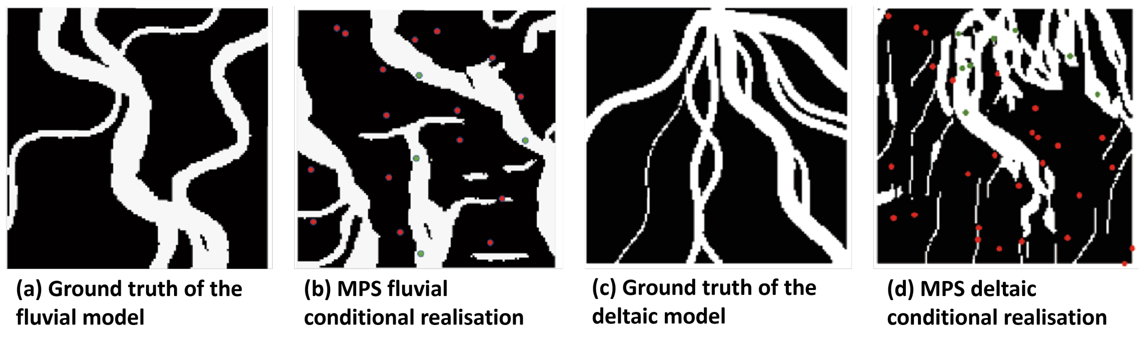

(a) Ground truth of the fluvial model
(b) MPS fluvial conditional realisation
(c) Ground truth of the deltaic model
(d) MPS deltaic conditional realisation

FIGURE 2.12: An example of conditional facies models from MPS. Modified after Zhang et al. [2019b].

2.3 Machine Learning-based Geological Modelling Approach

The machine learning approach optimises a model by learning from data to fit specific tasks, like image synthesis, that is later extended to solve facies modelling challenges. Deep generative models successfully reproduce fluvial channel geometry and constant facies transition along channels [Laloy et al., 2017, 2018]. Like most machine learning-based approaches, the best result of a deep generative model is producing the same quality data as their training data. Therefore, one can not expect deep generative models to create a sedimentary system with complex facies transitions when only training them with regular objects. A machine learning-based model is only as good as the data used for training and testing. Earlier publications also attempt to condition GANs to well data, net-to-gross, and probability maps, blending observed data and conceptual geology [Chan and Elsheikh, 2019, Song et al., 2021b, Zhang et al., 2019b]. Though previous studies prove that deep generative models provide a good chance for creating complex facies models, applying them to real fields still needs further efforts to tackle their training, evaluation and conditioning flexibility challenges. This section reviews the success and drawbacks of recent deep generative model applications in the literature.

This section introduces the workflow of deep generative models and their widespread applications to facies modelling, including Variational Auto-Encoder (VAE) and Generative Adversarial Networks (GAN). Deep generative models use similar workflows for training and generation. Thus, the first subsection introduces general components of deep generative models' workflow. The second subsection reviews VAE and its applications to reservoir modelling. This subsection presents the Encoder-Decoder model and the Auto-Encoder first, though most are not generative models because they have no distributional assumptions on data generation and purely project data to lower dimensional space. Introducing them here is because they share the same model structure as VAE and have some successful applications to reservoir modelling in earlier years [Canchumuni et al., 2017, Liu et al., 2019]. The last subsection broadly discusses GAN, its training difficulties, variants and applications to facies modelling in recent years.

2.3.1 Deep Generative Models Workflow

The training workflow of deep generative models commonly comprises three main components: data pre-processing, model structure and optimisation. This subsection introduces each element in general and reviews its typical compositions, respectively.

2.3.1.1 Data Pre-processing

Data pre-processing refers to techniques enhancing raw data quality for machine learning and data mining, including data cleaning, reduction, scaling, and transformation [Fan et al., 2021]. Data cleaning methods handle missing or noisy data by dropping, filling, or smoothing them. Data reduction refers to decreasing the data dimensions by selecting or extracting features. Data scaling and transformation modulate raw data to change the data range, distribution and type. Particularly for generative tasks, data scaling and transformation are essential steps before feeding raw data into the models. Data scaling can make the generative models easier to optimise, and data transformation can guarantee the data type is executable to generative models, e.g. GAN cannot directly process categorical data.

As for facies modelling, the modellers usually need to convert the facies to numerical data by data scaling and transformation methods. Facies represent certain rock types with contrasting properties, while most deep generative models require digital inputs. Several data transformation algorithms can efficiently convert data from category to numeric. Two common ways are integer encoding and one-hot encoding; each has an appetite for a particular type of categorical data and shortcomings in its conversion mechanism.

Integer encoding explicitly assigns an integer to each category (facies), and the value assignments usually obey a particular ordered relationship, also known as Ordinal encoding. For example, an integer encoder assigns 1, 2, 3 to represent size labels $small$, $medium$, and $large$. This method is straightforward and keeps the data volume unchanged. However, integer encoding could provide inaccurate relationships when categories don't appear in a single order [Hancock and Khoshgoftaar, 2020], which is the case for most facies associations. Many facies within the same model can be ordered in multiple ways based on different rock properties. Facies often have non-linear relationships with their rock properties, such as grain size, porosity, permeability etc. For example, facies with big grain sizes can have high and low permeability depending on cementing. Thus, ordering facies in one direction based on one property or linearly correlated properties is inappropriate when they don't have an apparent order due to the complex relationships between rock properties.

Another popular data transformation method for categorical data is one-hot encoding that replaces each category, e.g. facies variable, with a new binary vector [Niculescu-Mizil et al., 2009], the same as the indicator variable used in geostatistics (e.g. SIS). The new binary value, therefore, indicates whether a particular category is present at the current location. For instance, while processing a model composed of two colours, blue and red, a one-hot encoder produces a binary vector that represents blue and red as a vector of length two because there are two different colour types. The binary vector [1,0] means blue, while [0,1] denotes red. Because the one-hot encoder encodes each feature as a binary vector whose length equals the number of categories, it may cause the data volume to inflate and make training challenging if you have many categories. Indeed, too many facies is not only an issue for machine learning-based methods but also for all facies modelling tools, e.g. geostatistics [Strebelle, 2021].

Then, the modellers may need data scaling techniques to map data to a specific range (commonly a small range) and/or change the distribution shape as the next step of data transformation in the pipeline. The data scaling is often necessary when data have a varying or large range because a bigger value tends to give the attribute more 'weight' during a gradient-based optimisation, impacting the learning efficiency [Han et al., 2022]. Two popular data scaling methods in earlier publications are min-max normalisation and standardisation. Min-max normalisation, also known as rescaling, refers to linearly mapping data from its original range to a new range, which preserves the actual relationship between data values but may produce out-boundary results when new data locates outside the original scope [Han et al., 2022]. Standardisation, also called z-score normalisation, centres data by calculating the difference between data value and mean value and then is divided by the standard deviation. Though it reduces the effect of 'outliers' [Han et al., 2022], standardisation changes the original data distribution to Gaussian distribution, which may be incorrect; for example, the channelised pattern of fluvial systems is non-Gaussian.

2.3.1.2 Model Structure

A Deep generative model generally contains one or more neural networks-based models that consist of various layers, including fully connected layers, pooling layers, convolutional layers, normalisation layers, activation functions, and so on. This subsection overviews those elements composing neural networks in deep generative models.

A fully connected layer, also known as the dense layer, describes a vector whose elements, called 'neurons', are all connected to every neuron in the following vector [Ramsundar and Zadeh, 2018]. Figure 2.13 illustrates a fully connected layer that calculates its outputs by multiplying a 2D weight matrix and adding a bias vector (see equation 2.3).

$$y = Wx + b \tag{2.3}$$

where y is the output vector, x is the input vector, W is the weight matrix whose size equals the output size times the input size, and b is the bias vector. Elements in both W and b are learnable parameters that a training process can update their values. Fully connected layers exploit all information provided to predict values in the output vector but often raise learning issues, such as overfitting, due to their heavy, in terms of a large amount of, learnable parameters.

Overfitting is a common issue in neural networks, which refers to a model performing perfectly on training data but badly on test data. To prevent overfitting, people often apply

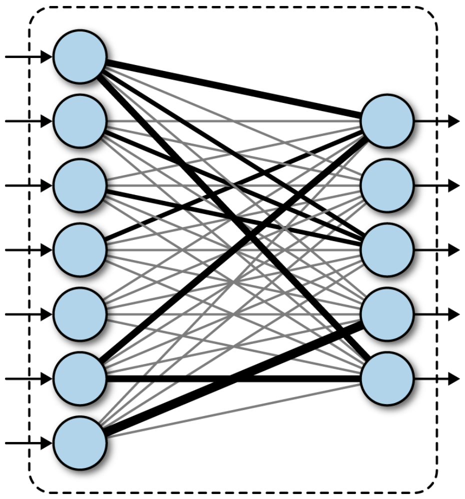


FIGURE 2.13: A schematic diagram of a fully connected layer. From Ramsundar and Zadeh [2018].

strategies, for example, early stopping, dropout, regularisation, etc., when they train neural networks [Ramsundar and Zadeh, 2018]. Early stopping means ceasing the training process before the model overfits. In practice, the modellers often save intermediate results during training and select a good model with the help of a validation set. Dropout is a process that randomly freezes some neurons and only updates the rest at each training iteration, preventing neurons from overly co-adapting [Srivastava et al., 2014]. The regularisation penalises the model based on its parameters' variance, resulting in a simpler model with a reduced variance while not increasing the bias too much [Goodfellow et al., 2016]. However, deep learning research showed larger models could perform better with increased model complexity, making the 'bias-variance trade-off' debatable [Nakkiran et al., 2021]. Nakkiran et al. [2021] presented that many deep learning models' performance first gets worse but then gets better with the increase of model size, called the double-descent phenomenon (See Figure 2.15).

Normalisation layers standardise the input to accelerate and stabilise the training process by reducing the internal covariate shift, which refers to the change in the input distribution to a learning system [Ioffe and Szegedy, 2015]. Gradients vanishment and explosion are two common problems of using gradient-based optimisers to train deep learning models. During training, gradient explosion happens if the gradients of weights keep gaining along a direction, making the training process unstable. The gradient vanishment, on the other hand, happens when the gradients of weights keep descending and approach zero, resulting

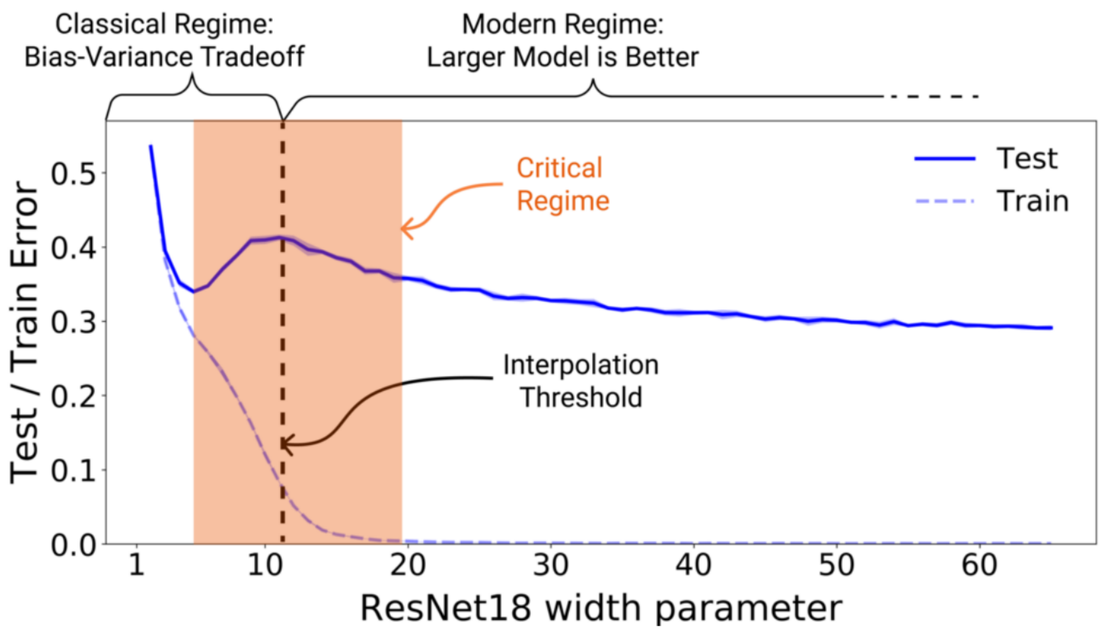


FIGURE 2.14: An example of the double-descent phenomenon. ResNet18 performs first worse and then better on the test set with the increased size of the width parameter. From Nakkiran et al. [2021].

in non-convergent behaviour.

Popular normalisation layers include batch normalisation, layer normalisation and instance normalisation [Ba et al., 2016, Ioffe and Szegedy, 2015, Ulyanov et al., 2016]. They apply the same feature normalisation but compute based on different pixel sets (see Figure 2.15) using Equation 2.4 [Wu and He, 2018]

$$\hat{x}_i = \frac{1}{\sigma_i}(x_i - \mu_i) \tag{2.4}$$

where $\hat{x}_i$ is the normalised value of input pixel x indexed i, μ_i is the mean of the pixel set, and σ_i is the standard deviation of the pixel set. The calculations of μ_i and σ_i are given by Equation 2.5 and 2.6 [Wu and He, 2018]

$$\mu_i = \frac{1}{m}\sum_{k \in S_i} x_k \tag{2.5}$$

$$\sigma_i = \sqrt{\frac{1}{m}\sum_{k \in S_i}(x_k - \mu_i)^2 + \epsilon} \tag{2.6}$$

where S_i is the pixel set, m is the size of the pixel set, k is the index of pixels in S_i, and ϵ is a small constant.

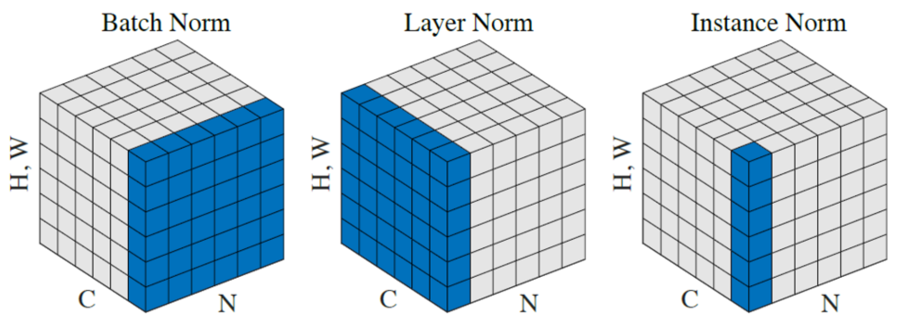


FIGURE 2.15: A schematic diagram of popular normalisation layers that process 4D feature tensors visualized in 3D. The blue pixels are the set of pixels used in calculating μ_i and σ_i in Equation 2.4. C denotes the channel dimension in the feature tensor. Please note the term 'channel' here is a dimension in tensor instead of the fluvial channel in geology. N denotes the batch dimension in the feature tensor. H, W are spatial axes representing height and width dimensions. From Wu and He [2018].

Activation functions transfer neurons' outputs to another value domain linearly or non-linearly to decide if a neuron is activated. Linear activation (see Figure 2.16 a) operates a linear transformation on its input using Equation 2.7,

$$f(x) = ax + b \tag{2.7}$$

where x is the input value, a and b are constant coefficients. Non-linear activation introduces non-linearity into neural networks, and using non-linear activations means that neural networks can capture non-linear patterns. Selecting activations depends on the task and layer position. The last activation function of the model needs to map its inputs to the model output range determined by the tasks, for example, numerical values for the image synthesis or categories for the image classification. This section lists five popular nonlinear activations in deep generative models: Sigmoid, Tanh, ReLU, Leaky ReLU and Softmax.

Little [1974] introduced the Sigmoid, also known as the logistic function, by analogy with spin systems, converting the input into a range between 0 and 1 based on the given Equation 2.8 [Han and Moraga, 1995]

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.8}$$

where x is the input value and e is the Euler number. The sigmoid function is differentiable, bounded and non-linear, making it a good choice of activation in neural networks to discover complex patterns, especially for the binary classification task. However, the two ends of the sigmoid activation tend to flatten, called sigmoid saturate, meaning a limited response to its input value variation resulting in vanishing gradients (see Figure 2.16 b). Another undesirable feature of the sigmoid function is its non-zero-centred nature, making the optimisations driven by gradient-based algorithms move along a single direction and slowing down convergence speed [Datta, 2020].

Hyperbolic tangent function denoted as Tanh is a popular non-linear and zero-centred activation, having a better training performance than the sigmoid function [Datta, 2020]. The Tanh converts its input into a range between -1 and 1 by the given Equation 2.9,

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.9}$$

where x is the input value and e is the natural constant. Although Tanh eases the pain of the non-zero-centred problem in model optimisation, it still suffers the saturated ends (see Figure 2.16 c), which means the model optimisation has the risk of vanishing gradients.

Rectified Linear Unit, denoted as ReLU, takes only the positive part of its input by the given Equation 2.10 [Jarrett et al., 2009, Nair and Hinton, 2010]

$$f(x) = \begin{cases} x, x > 0 \\ 0, x \leq 0 \end{cases} \tag{2.10}$$

where x is the input value. ReLU partly resolves the saturation problem at the positive end (see Figure 2.16 d) and is computationally faster than sigmoid and Tanh due to its simple calculation [Datta, 2020]. ReLU pushes its input to zero at the negative side, and therefore, the gradients become zero all the time, causing the units never to get activated, called the dying ReLU problem [Maas et al., 2013].

Maas et al. [2013] came up with a variant of ReLU, called leaky ReLU, to tackle the dying ReLU problem by giving ReLU a small slop at the negative part using the Equation 2.11

$$f(x) = \begin{cases} x, x > 0 \\ \alpha x, x \leq 0 \end{cases} \tag{2.11}$$

where x is the input value and α is a small constant gradient. He et al. [2015] further developed a variant of leaky ReLU, called parametric ReLU, by changing α from a small constant value to a learnable parameter. Leaky ReLU, as well as parametric ReLU, is a zero-centred but non-bounded function (see Figure 2.16 e), making it undesired to be the last activation function for many cases that require an output range. For example, many classification tasks need the model to predict a value between 0 and 1.

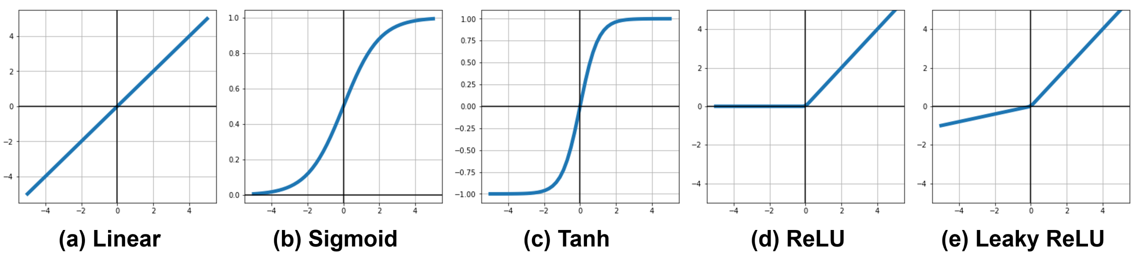


FIGURE 2.16: Examples of popular activation functions. (a) Linear. (b) Sigmoid. (c) Tanh. (d) ReLU. (e) Leaky ReLU

Softmax is a popular activation function mainly used for multi-class classification tasks, whose output is the relative probability of each class. Softmax projects its inputs to values ranging from 0 to 1 and assures the sum of all class probabilities at a certain location equals 1 using Equation 2.12 [Bridle, 1990]

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}} \tag{2.12}$$

where x is an input vector containing multiple elements, i and j are the index of the element in x, N is the total number of elements in the vector x, and e is the natural constant.

The pooling layer is a downsampling technique that subsamples inputs to scale down the spatial size, which decreases computational cost by reducing learnable parameters, allows convolutional neural networks to capture hierarchical patterns, and improves local translation invariance to prevent overfitting [Gholamalinezhad and Khosravi, 2020]. Local translation invariance states that most pooled output values remain unchanged when slightly translating the input, which is useful if the modellers are more concerned with the presence of a feature than its location [Goodfellow et al., 2016]. Two classical pooling operators are average pooling and max pooling, which divide their input into smaller regions to produce a lower representation. The difference is that average pooling calculates the average/mean values while the max pooling operator uses the maximum values to represent the regions.

In contrast, an inverse operation of pooling is upsampling that maps its input into a larger size, which many deep generative models require, e,g, GANs and VAEs. An upsampling operator expands data by evenly inserting units between original data points. Depending on the upsampling algorithm, a new unit can have the same value as its nearest neighbour or infer its value by an interpolation function, such as linear interpolation. Most classical upsampling layers, as well as pooling layers, have no learnable parameters and, therefore, add limited computational cost to the models.

A convolutional layer scans data using a set of learnable matrices with smaller spatial sizes, called kernels, to highlight if the patterns in the kernels exist (see Figure 2.17 a). A kernel needs to scan input data in different channels, compositing a block of matrix called a filter whose number equals the convolutional layers' output size in channel dimension. In practice, the convolution operator unrolls its input and output into vectors and uses a sparse matrix to represent the kernels, making the forward and backward calculations easier (see Figure 2.20). Convolutional layers have fewer learnable parameters than fully connected layers because the kernels' size is usually much smaller than the input data size, which benefits model training. The convolutional layer also exhibits some degree of local and deformation invariance because of its three features: the local receptive field, the shared weights and the sub-sampling nature, especially when paired with pooling layers [LeCun et al., 1998].

A convolutional layer links each neuron to smaller regions of its input called local receptive fields (see Figure 2.18). The first convolutional layer's receptive field equals its kernel size if it directly processes the input data. The kernels highlight patterns' existence, allowing the neurons to extract local features, such as edges. By stacking convolutional layers, kernels can compose high-order features, and the neurons in a deeper convolutional layer have a larger receptive field, which makes the subsequent convolutional layers can identify more complex patterns, for example, meandering channels. When a model has a single path from input to output, a deeper layer's receptive field is a function of its kernel size and all shallower layers' kernel sizes and strides (see Equation 2.13 from [Araujo et al.,

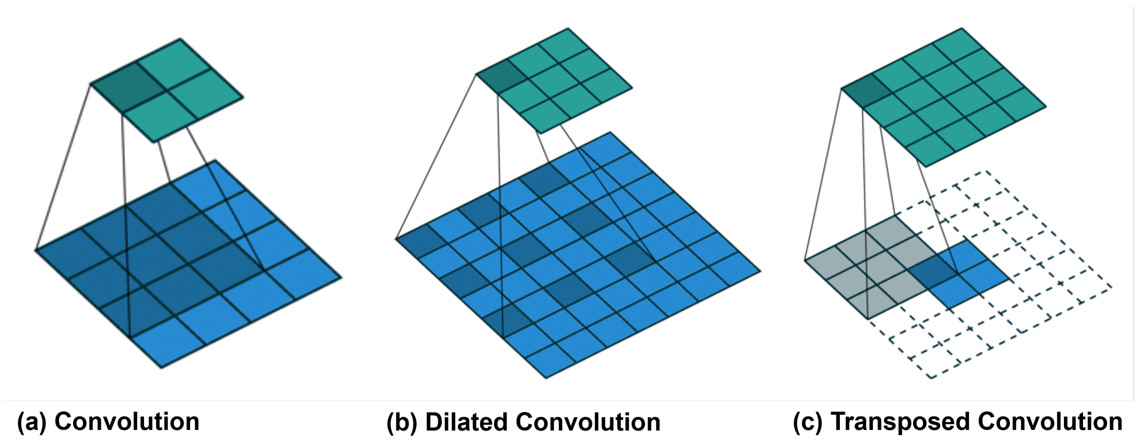


FIGURE 2.17: A schematic diagram of different convolutions. (a) A convolution operator. (b) A dilated convolution operator. (c) A transposed convolution operator. Modified after Dumoulin and Visin [2016].

2019]).

$$r_0 = \sum_{l=1}^{L}((k_l - 1)\prod_{i=1}^{l-1} s_i) + 1 \tag{2.13}$$

where r_0 is the receptive field, l denotes a layer, L is the total number of layers, k_l denotes the kernel sizes of layer l, s_i denotes shallower layers' strides, which means the distance that kernels move along a direction. According to Araujo et al. [2019], the modellers can regard pooling layers as having a kernel whose size equals the number of input units used to produce an output unit. Activation and normalisation layers are commonly not involved in receptive field calculation or change the receptive field of the whole network [Araujo et al., 2019]. Alternatively, the modellers can calculate the receptive field layer-by-layer using Equation 2.14 from [Araujo et al., 2019].

$$r_{l-1} = s_l \cdot r_l + (k_l - s_l) \tag{2.14}$$

where r_{l-1} is the receptive field on the input of layer l, s_l is the stride of layer l, r_l the receptive field on the output of layer l, and k_l is the kernel size of layer l.

Convolutional kernels produce output neurons in the form of layers called feature maps by scanning the whole inputs with the same weights. One feature map indicates a certain feature's existence through every region of the whole input. In general, a higher unit value in a feature map means a feature represented by the kernels is more likely to exist in its receptive field. This shared weight design makes convolution operators less sensitive to

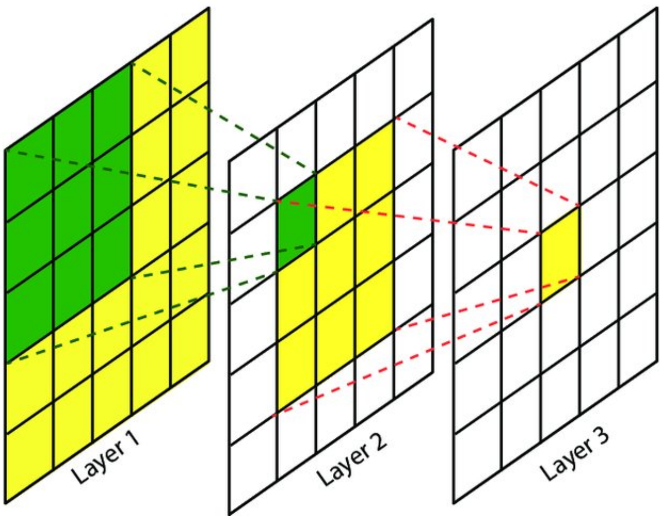


FIGURE 2.18: A schematic diagram of the receptive field of a convolutional neural network. From Lin et al. [2017].

the location while focusing more on feature detection.

A convolutional layer has a subsampling nature representing a patch of input by a unit, which can reduce the spatial size and resist slight deformation. Without padding, this operation makes units at the boundary used less than units at the middle. To tackle this problem, the modellers can insert units, normally zero values, at the outer boundary of input data. This process, known as padding, influences the output size that is a function of input size, kernel size, stride and padding (see Equation 2.15 from [Dumoulin and Visin, 2016]).

$$o = [\frac{i + 2p - k}{s}] + 1 \tag{2.15}$$

where o is the output size, i is the input size, p is the padding size, k is the kernel size, and s is the stride. When the stride is bigger than one, a convolutional layer can efficiently reduce the spatial size of input data. The value of each unit on the feature map results from the information inside the receptive field on the input data. Therefore, a slight transformation in the input data won't change much on the feature maps. Particularly when a pooling layer follows the convolution, it subsamples and smooths the feature maps, reducing the transformation sensitivity of outputs, which helps tackle irregular data with a shape variation, such as fluvial channels.

Dilated convolution is a convolution operator that broadens the kernel by inserting empty elements between kernel elements whose occurrence is controlled by a dilation rate [Yu and Koltun, 2015]. The dilated convolution layers have a larger receptive field than a standard convolution layer when they have the same kernel sizes (see Figure 2.17 b). A 3×3 kernel with a dilation rate equal to 2, which means scanning every second unit within the expanded kernels, has a receptive field of 5×5, capturing a bigger pattern at the same computational cost as standard convolution using a 3×3 kernel. The new parameter, the dilation rate, widens the kernel size and impacts the output size. The calculation of dilated convolutional layers' output size is given by [Dumoulin and Visin, 2016],

$$o = [\frac{i + 2p - k - (k-1)(d-1)}{s}] + 1 \tag{2.16}$$

where o is the output size, i is the input size, p is the padding size, k is the kernel size, s is the stride, and d is the dilation rate. Though a dilated convolution can have a bigger receptive field without increasing the computational workload by inserting empty elements, the standard dilated convolution suffers a 'gridding effect' (see Figure 2.19) due to the loss of neighbouring information [Wang et al., 2018a]. Thus, the modellers need to carefully tune the hyper-parameters when designing a dilated neural network-based model to prevent the increased efficiency from resulting in poor performance.

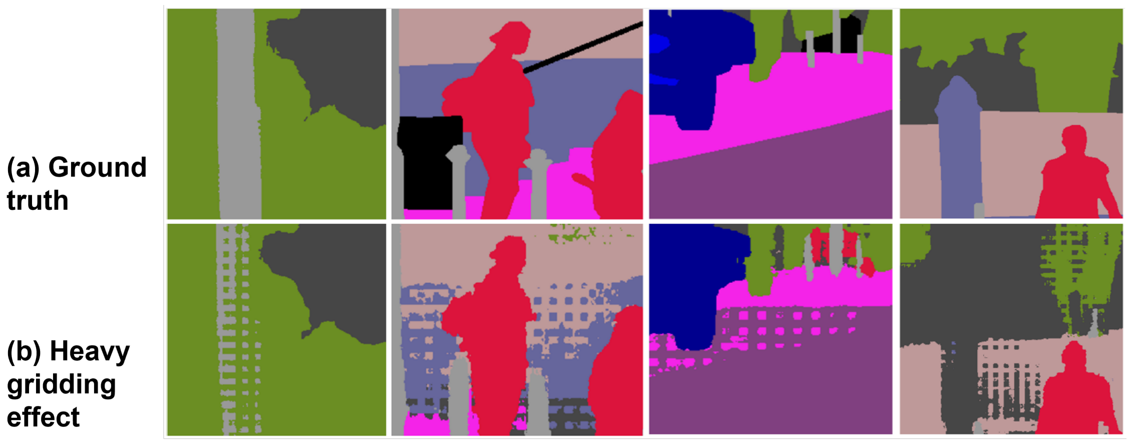


FIGURE 2.19: An illustration of gridding effect. (a) Ground truth of the semantic segmentation maps. (b) A GAN suffers a heavy 'gridding effect'. Modified after Wang et al. [2018a].

Transposed convolution, also known as fractionally-strided convolution, swaps the matrix multiplication during forward and backward calculation in standard convolution to upsample the input data [Dumoulin and Visin, 2016]. In practice, the transposed convolution operator transposes the sparse matrix in the standard convolution (see Figure 2.20).

This operator provides an alternative way of upsampling (see Figure 2.17 c), which is particularly useful for deep generative models because they require both downsampling and upsampling. The output size of transposed convolution is given by [Dumoulin and Visin, 2016],

$$o = s(i - 1) + k - 2p \tag{2.17}$$

where o is the output size, i is the input size, p is the padding size, k is the kernel size, and s is the stride.

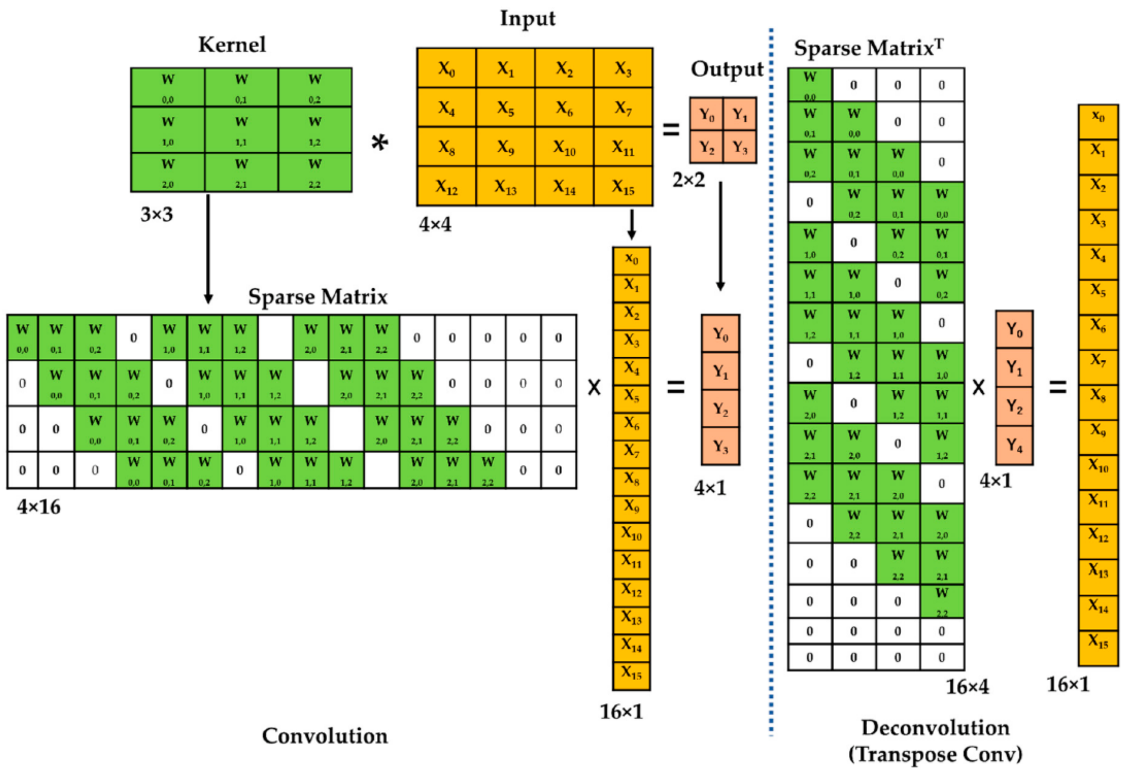


FIGURE 2.20: A schematic diagram of sparse matrix multiplications in convolution and transposed convolution. From Islam and Kim [2019].

2.3.1.3 Optimisation

To date, deep generative models' optimisation relies on backpropagation, which is an algorithm that automatically calculates every parameter's gradient for the gradient-based optimiser [Rumelhart et al., 1985]. The backpropagation algorithm preserves all intermediate results during the forward calculation from inputs to outputs and then backpropagates the error from the loss function based on the chain rule to re-calculate the gradients (adjust the neuron weights). The chain rule is a formula to calculate the derivative of a complicated function composed of a set of simple equations, given by Equation 2.18 for $f(g(x))$

[Guichard et al., 2020]

$$\frac{df}{dx} = \frac{df}{dg} \cdot \frac{dg}{dx} \tag{2.18}$$

Then a gradient-based optimizer, for example, stochastic gradient descent [Robbins and Monro, 1951], updates all the learnable parameters to minimize the objective function by Equation 2.19 [Ruder, 2016]

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta) \tag{2.19}$$

where θ is the learnable parameters, η is the learning rate that is a hyperparameter to decide the impact of the gradient on one update step, and $\nabla_\theta J(\theta)$ is the gradient of the objective function to the parameters. Sutton [1986] highlighted two problems associated with the gradient descent mechanism that slow down optimising the machine learning models. One problem is the local minimum trap due to the steeper gradient in one direction than others, and the other problem is the optimiser tends to modulate identified useful units instead of exploring others to handle new patterns [Sutton, 1986].

Qian [1999] introduced a new term, momentum, to gradient descent algorithms by analogy with the Newtonian particles' movements to accurate the convergence speed. The current machine learning community widely adopts the implementation of stochastic gradient descent (SGD) with momentum whose calculation given by Equation 2.20 [Ruder, 2016]

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_\theta J(\theta) \\ \theta &= \theta - v_t \end{aligned} \tag{2.20}$$

where θ is the learnable parameters, η is the learning rate, $\nabla_\theta J(\theta)$ is the gradient of the objective function to the parameters, v_t and v_{t-1} are the velocities at current and last time steps, respectively, and γ is a constant hyperparameter of momentum term. SGD is sensitive to the learning rate and uses the same learning rate to update all parameters, influencing the convergence speed of learning rare features from sparse data [Kingma and Ba, 2014].

Kingma and Ba [2014] invented an adaptive moment estimation, Adam, to accelerate the optimisation by computing an adaptive learning rate for every parameter. Adam combines

AdaGrad and RMSprop's advantages, which decay the learning rate based on the past gradients for every parameter [Kingma and Ba, 2014]. Adam updates parameters by Equation 2.21 [Ruder, 2016]

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \\ \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t} \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t \end{aligned} \tag{2.21}$$

where t denotes the step index, m refers to the first-moment term (mean) of the gradients, v refers to the second-moment term (variance) of the gradients, g is the gradients, β_1 and β_2 are constant hyperparameters controlling the decay rates of the first and second moments, respectively, $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moments, respectively, θ is the learnable parameters, η is the learning rate, and ϵ is a small constant preventing the denominator becoming zero. Though Adam converges fast and has equal or better performance than SGD and other adaptive gradient methods, many state-of-art models still use SGD with momentum to achieve their best results as Adam seems not as good as SGD with momentum regarding the generalisation performance [Kingma and Ba, 2014, Loshchilov and Hutter, 2017, Wilson et al., 2017].

Loshchilov and Hutter [2017] claimed that L2 regularisation is not equivalent and works less effectively than weight decay in Adam by comparing those two classical methods of improving model generalisation and, therefore, proposed a variant of Adam with decoupled weight decay called AdamW. Hanson and Pratt [1988] came up with the weight decay that is given by the Equation 2.22 [Loshchilov and Hutter, 2017]

$$\theta_{t+1} = (1-\lambda)\theta_t - \alpha \nabla f_t(\theta_t) \tag{2.22}$$

where θ is the parameter, t is the step index, λ is the weight decay factor, α is the learning rate, and $\nabla f_t(\theta_t)$ is the gradient of parameter. In contrast, the classical implementation of L2 regularisation adds a penalty term when calculating the gradients, given by Equation

2.23

$$g_t = \nabla f_t(\theta_{t-1}) + \lambda\theta_{t-1} \tag{2.23}$$

where g is the gradients of parameters, t denotes the step index, f denotes the loss function, θ is the parameter, and λ is the L2 regularisation factor. This implementation in standard SGD (combine Equation 2.19 and 2.23) is equivalent to applying weight decay. However, L2 regularisation (Equation 2.23) involves the penalty term into the gradient that is later used to calculate the first-moment and second-moment terms in Adam (see Equation 2.21), leading to the inequivalence [Loshchilov and Hutter, 2017]. AdamW decouple the weight decay from the gradient calculation during Adam optimisation that updates parameters by Equation 2.24 [Loshchilov and Hutter, 2017]

$$\theta_t = \theta_{t-1} - \eta_t(\frac{\alpha\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda\theta_{t-1}) \tag{2.24}$$

where θ is the parameter, t denotes the step index, η is a scaling factor, α is the learning rate, $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moments, respectively, ϵ is a small constant, and λ is the weight decay factor.

2.3.2 Encoder-Decoder Models

Encoder-Decoder models are end-to-end approaches that learn to map data from one domain to another, also known as the Sequence to Sequence model introduced by Sutskever et al. [2014]. An Encoder-Decoder architecture commonly contains two neural network-based models (also can be regarded as a single networks with two components), the Encoder and the Decoder. The encoder compresses the data from a high-dimensional domain into a lower latent representation. At the same time, the Decoder transfers data from the latent state to the target domain that may differ from the input data domain. For example, Sutskever et al. [2014] applied an Encoder-Decoder model to translate English to French.

As for reservoir modelling, Liu et al. [2019] used an Encoder-Decode Model as their transform net and a well-trained VGG-16 classifier as a feature extraction tool to generate facies models (see Figure 2.21). They proved their deep learning-based method, CNN-PCA, outperformed traditional methods by comparing their way with the Principal Component

Analysis (PCA) and the Optimisation-Based PCA (O-PCA) for realisation parameterisation and flow simulation history matching. The deep learning-based approach efficiently parameterises (in terms of dimension reduction) the facies model and pre-images them successfully with a high image quality showing semi-straight channelised shapes without a blurred boundary around channels.

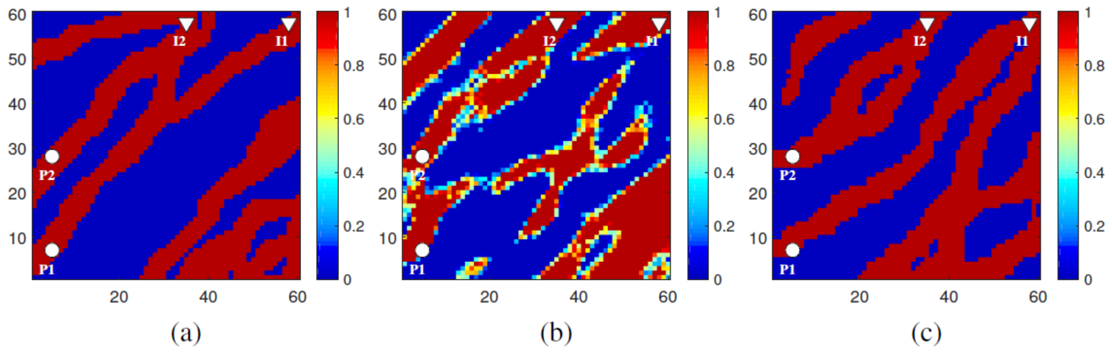


FIGURE 2.21: An example of Encoder-Decoder model generations compared to ground truth and a benchmark solution. Modified after Liu et al. [2019]. (a) the ground truth realisation. (b) a conditional realisation from O-PCA. (c) a conditional realisation from CNN-PCA (a deep learning-based method with an Encoder-Decoder model).

2.3.2.1 Auto-Encoder (AE)

An Auto-Encoder is a particular case of the Encoder-Decoder model, whose input and output domains are the same [Goodfellow et al., 2016]. It aims to reproduce its inputs at the output layers (see Figure 2.22), which works as a non-linear solver of reducing data dimension in its earlier applications [Goodfellow et al., 2016, Kramer, 1991]. Thus, AE can efficiently compress non-linear data while free of the pre-image problem which is a typical challenge of projecting compressed data back to the original data space for many dimension reduction algorithms, such as PCA. A further application of AE is denoising, also known as Denoising Auto-Encoder (DAE), which uses pairs of corrupted and clean data instead of the same data as inputs and outputs [Goodfellow et al., 2016]. This change broadens AEs application to handle many end-to-end tasks, for example, image denoising that has achieved notable successes in denoising seismic images [Mandelli et al., 2019].

Canchumuni et al. [2017] applied Auto-Encoder to parameterise facies model and integrated their trained Encoder and Decoder models into a history matching framework based

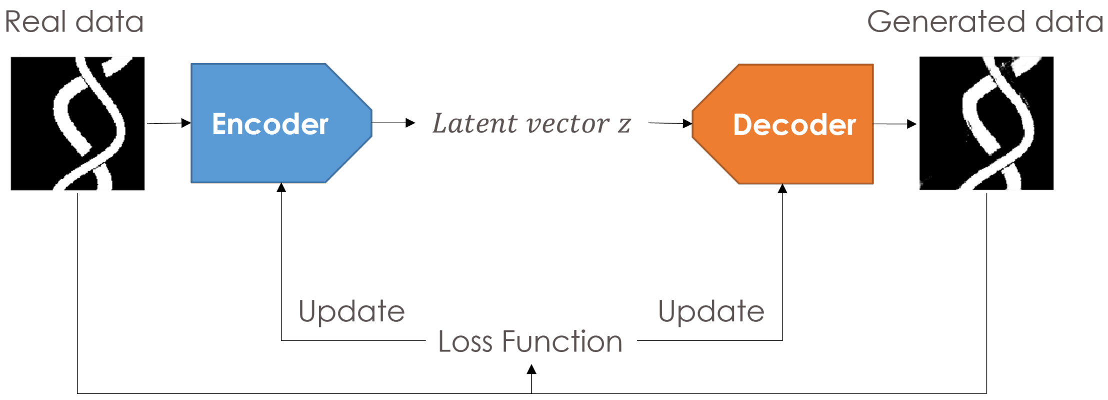


FIGURE 2.22: A schematic diagram of the Auto-Encoder (AE) workflow.

on an ensemble smoother with multiple data assimilation (ES-MDA). Auto-Encoder successfully reconstructs the facies model with minor errors at geo-bodies' boundary (see Figure 2.23). As their study case is PUNQ-S3, they created their training dataset using Multi-point Statistics [Remy et al., 2009] and, therefore, their Auto-Encoder can only capture the spatial correlation represented by the MPS.

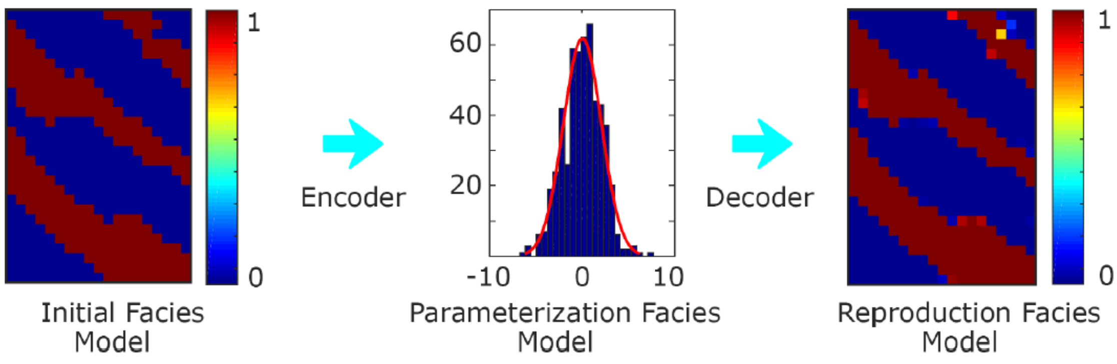


FIGURE 2.23: An example of Auto-Encoder generations showing the initial and reproduced facies models. Modified after Canchumuni et al. [2017].

2.3.2.2 Variational Auto-Encoder (VAE)

Variational Auto-Encoder (VAE), a variant of Auto-Encoder, works by finding the multivariate latent distribution of the given datasets [Kingma and Welling, 2013]. VAE model also contains an Encoder network and a Decoder network. However, instead of directly creating a latent vector as the Decoder's input, the Encoder produces a multivariate distribution, including a mean vector and a standard deviation vector, which generate an input vector for the Decoder by sampling on the latent distribution. Figure 2.24 show a schematic diagram of VAE. Unlike AE, which updates its parameters by only calculating the difference between original and reconstructed samples, VAE also needs to calculate the Kullback–Leibler divergence (KL divergence) of the approximate from the true posterior [Kingma and Welling, 2013]. This KL divergence term forces the latent distribution to approach a prior distribution, for example, a multivariate Gaussian distribution.

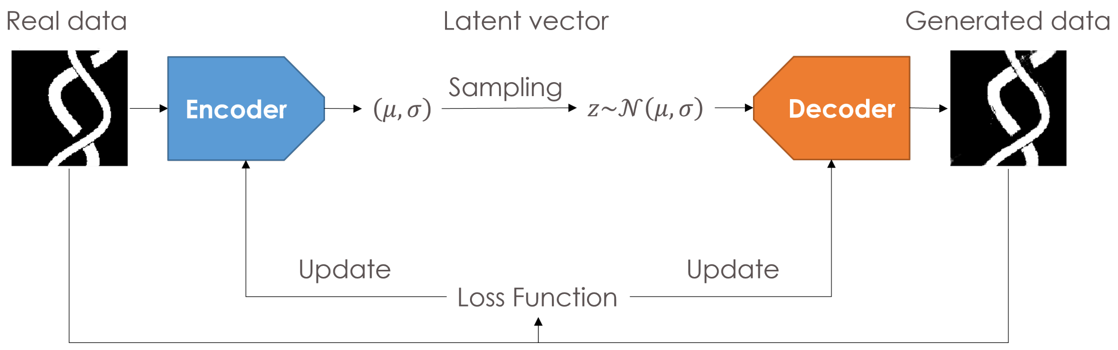


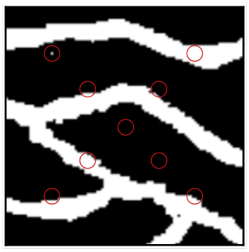
FIGURE 2.24: A schematic diagram of the Variational Auto-Encoder (VAE) workflow.

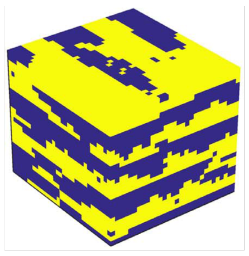
Laloy et al. [2017] implemented a VAE to parameterise a set of binary facies models simulated by MPS, including 2D unconditional models, 2D conditional models and 3D unconditional models (see Figure 2.25). Compared to the training dataset, VAE captures the sinuous patterns in 2D and reproduces 3D models that resemble the 3D data from MPS. However, the conditional models from VAE have some mismatches in conditioning data because they didn't use any conditioning technique in their workflow and only relied on the training dataset composed of conditional simulations from MPS. This conditioning method is inflexible in handling fields with different numbers and locations of wells, which requires re-simulating the training dataset. Both the training dataset and trained VAE become disposable, increasing the time and computational cost of deep learning-based approaches.

2.3.3 Generative Adversarial Networks (GAN)

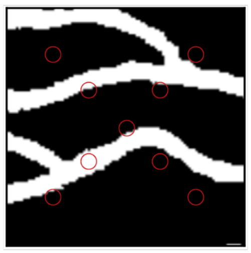
Generative Adversarial Networks (GAN) refer to competitive learning between two neural network-based models called the generator and the discriminator [Goodfellow et al., 2020]. Figure 2.26 shows a typical GAN workflow that plays a minimax game between the generator and the discriminator to update their parameters using the loss function given

(a) 2D unconditional model from MPS


(b) 2D conditional model from MPS


(c) 3D unconditional model from MPS

(d) 2D unconditional model from VAE


(e) 2D conditional model from VAE

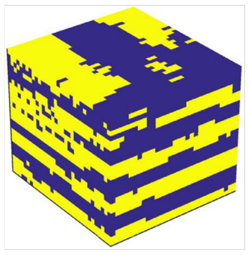

(f) 3D unconditional model from VAE

FIGURE 2.25: An example of VAE results compared to the training datasets produced by MPS. The red circles are well locations. Modified after Laloy et al. [2017].

by Equation 2.25 [Goodfellow et al., 2020]

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{x \sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim P_z(z)}[\log (1 - D(G(z)))] \qquad (2.25)$$

where $V(D, G)$ denotes the GAN loss function, $G(\cdot)$ and $D(\cdot)$ are the generator and the discriminator, respectively, z denotes the input vector, x denotes the real data, and $\mathbb{E}[\cdot]$ is the mathematical expectation. The generator projects a Gaussian (or Uniform) input vector z into a high-dimensional data domain $G(z)$, for example, images. At the same time, the discriminator aims to distinguish generated data $G(z)$ from real data x as a binary classifier. In practice, GAN labels real data x as one and generated data $G(z)$ as zero when calculating the loss value for the discriminator while labelling the generated data as one when calculating the loss for the generator, then compares the discriminator outputs with corresponding labels during training.

GAN training is notoriously challenging and unstable due to mode collapse, convergence failure, gradient explosion and vanishment. Section 2.3.1.2 has explained gradient explosion and vanishing gradients there. Mode collapse refers to GAN only creating one or a few samples instead of all types of data shown in the training dataset (see Figure 2.27 from

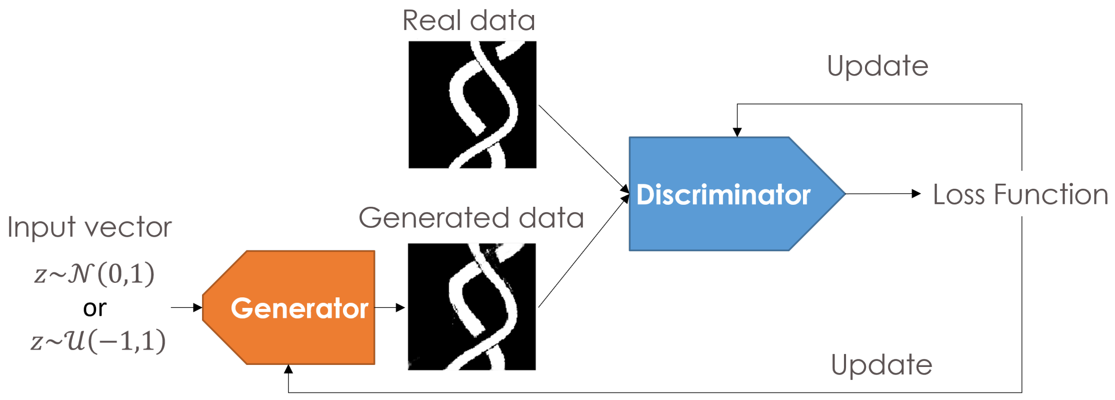


FIGURE 2.26: A schematic diagram of the Generative Adversarial Networks (GAN) workflow.

[Metz et al., 2016]). Avoiding mode collapse is, therefore, pretty influential in reservoir facies modelling because the modellers often aim to reproduce the model diversity of realisations to capture and quantify the subsurface uncertainty. Contrastingly, convergence failure leads to visually bad images, with poor replication of shapes in the training data. These issues attribute to the competitive nature of GANs. The discriminator improves at the cost of the generator getting a larger loss and vice versa, making it difficult to attain a balance point in the training process, called Nash equilibrium.

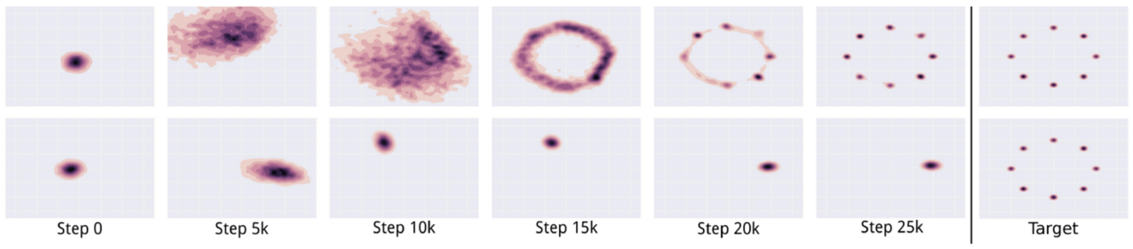


FIGURE 2.27: An illustration of mode collapse. The top row is a GAN free of mode collapse problem. The bottom row is a mode collapse case of GAN. From Metz et al. [2016].

To address the training difficulties, many publications improve GAN in different aspects of the GAN training workflow, for example, model structures, loss functions, and training tricks. Researchers published some empirical training tricks at an earlier stage as GAN is still not fully interpretable, which may be helpful in some cases but can't ensure performance and stability improvement. For example, Salimans et al. [2016] suggested using label smoothing when computing the loss values, which means using smoothed values, such as 0.1 and 0.9, instead of zero and one to label data. Those training tricks are easily adaptable to different GAN training workflows. In contrast, some researchers changed the model structures and loss functions, yielding a popular variant choice. For example, Radford et al. [2015] came up with a new GAN model structure called DCGAN, Arjovsky et al. [2017] replace the original loss function in Equation 2.25 with the Wasserstein distance, named WGAN. The following sections review popular GAN variants and their applications to facies modelling.

2.3.3.1 Deep Convolutional Generative Adversarial Network (DCGAN)

The deep convolutional generative adversarial network (DCGAN) model is a benchmark GAN structure broadly used in previous GAN applications as their models' architecture, including discriminators and generators [Radford et al., 2015]. Radford et al. [2015] provided a guideline to get a stable DCGAN by replacing pooling with convolutions and transposed convolutions, applying batch normalisations, removing fully connected layers, using ReLU and Tanh in the generator, and using Leaky ReLU in the discriminator (see Figure 2.28).

However, DCGAN has many variants containing somewhat different elements. For example, in some implementations, people still use fully connected layers in deeper architectures instead of a fully convolutional neural network [Teoh and Rong, 2022]. Odena et al. [2016] argued using transposed convolution caused the checkerboard artefacts (see Figure 2.29) and suggested using up-sampling followed by a convolution to avoid this problem.

As for facies modelling, Laloy et al. [2018] applied a variant of DCGAN, called spatial GAN, that uses a 2D or 3D spatial tensor instead of the vector as the generator input [Jetchev et al., 2016], to generate 2D and 3D binary facies model. The results show GAN can reproduce binary realisations resembling its training data (see Figure 2.30). However, as the training datasets used in their study are relatively simple geology, GANs' learning capability lacks a thoughtful exploration, resulting in wasting computational power. Their direct conditioning method relies on searching a pre-trained GAN's latent space, which converges slowly and even fails to converge.

Zhang et al. [2019b] further explored GAN learning 2D and 3D geomodels while honouring well observations. They used the DCGAN architecture to train their unconditional GAN and then optimised the latent vector of pre-trained GAN to honour well data (see

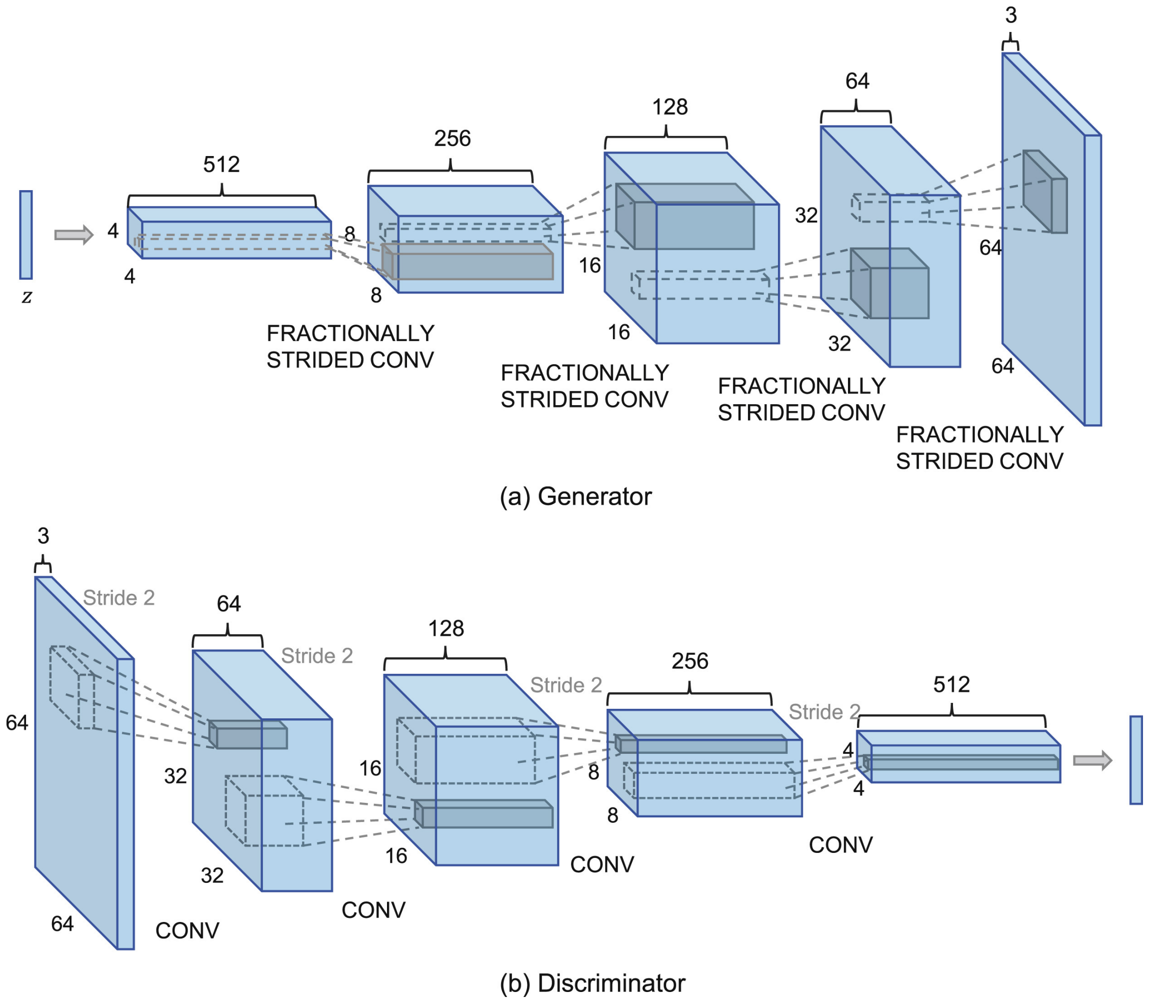


FIGURE 2.28: A schematic diagram of DCGAN. (a) the generator architecture. (b) the discriminator architecuture. z denotes the input noise vector. CONV is the abbreviation of the convolutional layer. From Hayashi et al. [2019].

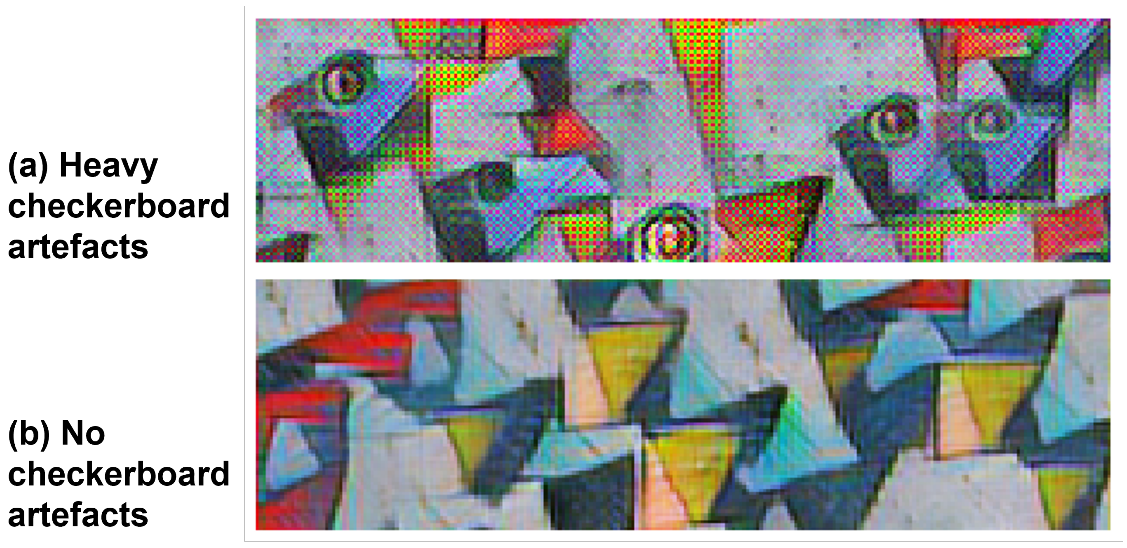

(a) Heavy checkerboard artefacts
(b) No checkerboard artefacts

FIGURE 2.29: An illustration of checkerboard artefacts. (a) A GAN suffers heavy checkerboard artefact. (b) A GAN is free of checkerboard artefacts. Modified after Odena et al. [2016].

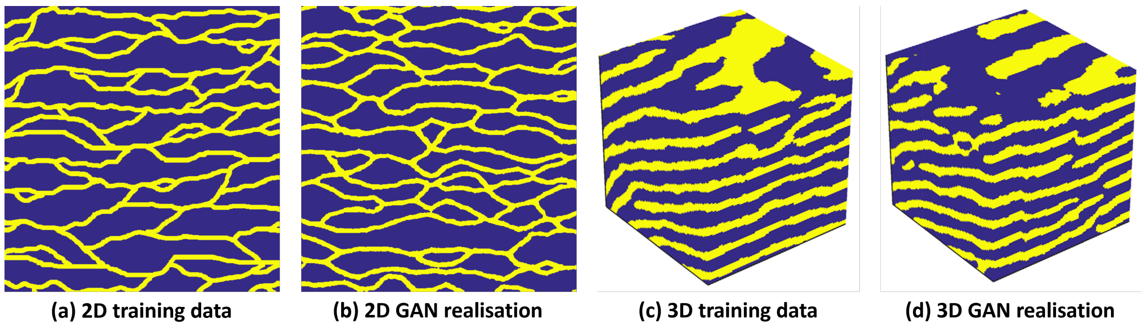


FIGURE 2.30: An example of GAN application to 2D and 3D facies modelling. Modified after Laloy et al. [2018].

Figure 2.31). They argued GAN outperformed MPS for creating conditional nonstationary realisation by visual comparisons (see Figure 2.12 b and 2.31 b). One bright feature of their conditioning method is the loss term calculating the distance between the mismatched well location and the nearest location (pixel/voxel) that has the corresponding (correct) facies instead of the difference between facies codes (please refer to Section 6.2 for the details). However, their conditioning method still relies on an optimisation approach to minimise mismatches at well locations.

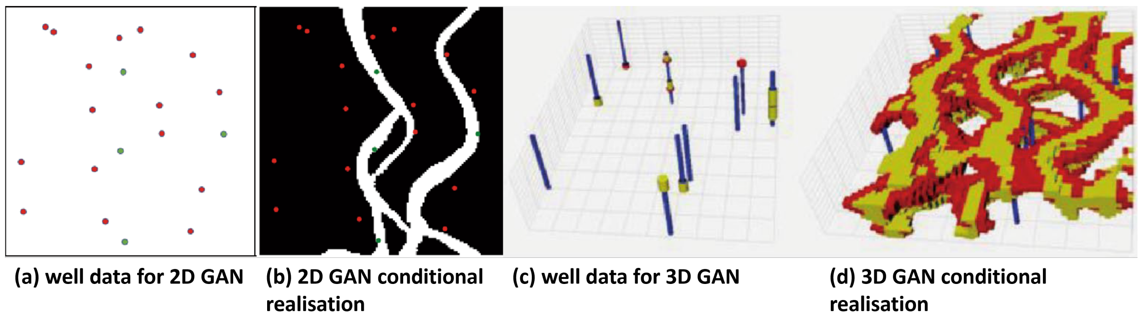

(a) well data for 2D GAN
(b) 2D GAN conditional realisation
(c) well data for 3D GAN
(d) 3D GAN conditional realisation

FIGURE 2.31: An example of GAN producing 2D and 3D conditional realisations. Modified after Zhang et al. [2019b].

2.3.3.2 Wasserstein Generative Adversarial Network (WGAN)

Arjovsky et al. [2017] claimed that using JS-divergence as the loss function (see Equation 2.25) causes gradients to vanish, and came up with Wasserstein GAN, which uses the Earth-Mover distance to replace JS-divergence as the loss function. WGAN updates its discriminator, called critic in WGAN, by Equation 2.26 and updates its generator by Equation 2.27

$$Loss_D = -\mathbb{E}_{x\sim P_x}[D(x)] + \mathbb{E}_{z\sim P_z}[D(G(z))] \tag{2.26}$$

$$Loss_G = -\mathbb{E}_{z \sim P_z}[D(G(z))] \tag{2.27}$$

where z denotes the latent vector, x denotes the real data, $G(\cdot)$ and $D(\cdot)$ are the sequential calculations of the generator and the discriminator in neural networks, and $E[\cdot]$ is the mathematical expectation. To improve stability, Arjovsky et al. [2017] empirically applied weight clipping on the discriminator's parameters, forcing values to be between -0.01 and 0.01.

A further improved version of WGAN, called WGAN-GP, replaces weight clipping with gradient penalty to force the discriminator to lie within the space of 1-Lipschitz functions [Gulrajani et al., 2017]. The gradient penalty is an extra term in the loss function to penalise the discriminator, whose calculation is given by Equation 2.28

$$GP = \lambda \mathbb{E}_{\tilde{x}}[(\|(\nabla D(\tilde{x}))\| - 1)^2] \tag{2.28}$$

where $\mathbb{E}[\cdot]$ is the mathematical expectation, ∇D is the gradient of the discriminator parameters, $\tilde{x}$ is a linear interpolation between real and fake data, and λ is a constant weight of gradient penalty.

Chan and Elsheikh [2019] applied WGAN to learn 2D binary channelised reservoirs and combined pre-trained WGAN with an inference network to condition GAN generations to well data. WGAN successfully learned the channel geometry (see Figure 2.32a), and the conditional generation also has a good match to observed data (see Figure 2.32b). Their conditioning method reuses the pre-trained neural networks, saving time in preparing datasets and training models. Instead of searching pre-trained GAN's latent vector, the inference network produces a new latent vector obeying Gaussian/uniform distributions that can also directly plugin existing model updating pipelines. However, as when searching latent space, training an inference network can be slow and lead to convergence failure (see more details in Section 6.2).

2.3.3.3 Conditional Generative Adversarial Network

The conditional generative adversarial network (conditional GAN) is a different type of GAN model, which uses extra information in and after training to constrain its generations. To the best scope of my knowledge, the first conditional GAN model published by Mirza

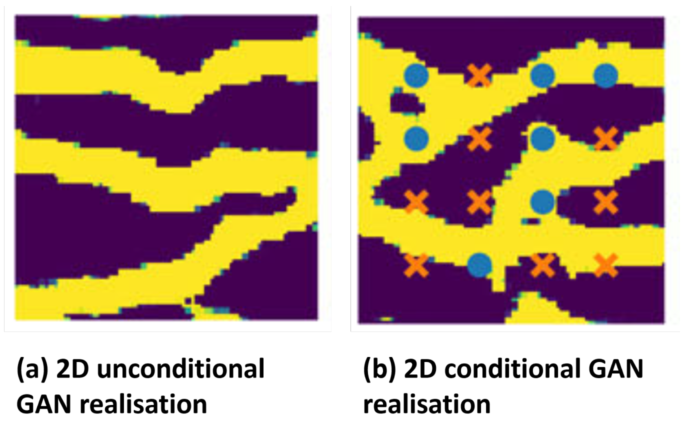


FIGURE 2.32: An example of WGAN producing 2D unconditional and conditional realisations. Modified after Chan and Elsheikh [2019].

and Osindero [2014] incorporates conditioning data into both the generator and discriminator as an extra input during training. The conditional GAN model, therefore, calculates the joint hidden representation of both data and conditions and updates its parameters by Equation 2.29 [Mirza and Osindero, 2014]

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{data}(x|y)}[\log D(x)] + \mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z|y)))] \tag{2.29}$$

where $V(D,G)$ denotes the conditional GAN loss function, $G(\cdot)$ and $D(\cdot)$ are the generator and the discriminator, respectively, z denotes the input vector, x denotes the real data, y is the condition and $\mathbb{E}[\cdot]$ is the mathematical expectation. Although Mirza and Osindero [2014] only shows a simple way of feeding conditioning data to GAN, this work encouraged later studies on developing higher-order interactions of conditioning data, for example, image-to-image translation models [Isola et al., 2017].

Image-to-image translation models predict images in a target domain based on given maps, for example, semantic segmentation maps [Isola et al., 2017]. One popular model, called $pix2pix$, uses convolutional neural networks to extract features from the conditioning data as the extra inputs of the generator's hidden layers [Isola et al., 2017]. Then Wang et al. [2018b] further develop $pix2pix$ model to tackle a high-resolution generation task, named $pix2pixHD$, by decomposing the generator into two sub-networks. One sub-network works as a global generator, and the other works as a local enhancer in which they are jointly trained to produce high-resolution images (see Figure 2.33).

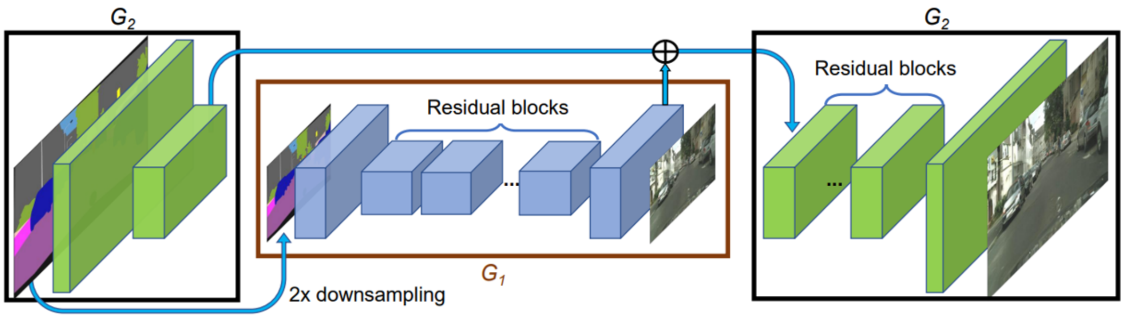


FIGURE 2.33: A schematic diagram of *pix2pixHD* model's generator architecture. G1 denotes the global generator. G2 denotes the local enhancer. From Wang et al. [2018b].

Park et al. [2019] developed a SPADE generator based on a conditional normalisation, Spatially-adaptive de-normalisation (SPADE), to replace the *pix2pixHD* model's encoder-decoder structure. SPADE uses convolutional neural networks to extract features from conditioning data to modulate (denormalise) the generator's normalised activations (outputs of normalisations, e.g. batch normalisation) [Park et al., 2019]. During training, the SPADE learns to create two tensors to scale and bias the corresponding elements of the generator's normalised activations (see Figure 2.34). The two learned tensors modulate the corresponding element values by Equation 2.30 [Park et al., 2019]

$$\gamma^i_{c,y,x}(m)\frac{h^i_{n,c,y,x} - \mu^i_c}{\sigma^i_c} + \beta^i_{c,y,x}(m) \tag{2.30}$$

where γ is the learned scaling parameter, β is the learned bias parameter, m is the conditioning data, h is the hidden layer before normalisation, μ is the mean value of activation in a certain pixel set, σ is the standard deviation of activation in a certain pixel set, i denotes the layer number of a deep convolutional network, and n, c, y, and x are the index of the batch, channel, height, and width dimension, respectively. Before feeding the conditioning data into SPADE, the SPADE generator downsamples it to adapt to every hidden layer block's size (see Figure 2.35). Compared to the *pix2pixHD* model, the SPADE generator decreases learned parameters by nearly half, favouring the training and reducing the chance of overfitting [Park et al., 2019].

Song et al. [2021a] presented successes in using conditional GAN techniques to force GAN generations to honour global features, well data and soft maps. Song et al. [2021a] encoded global features, such as orientation and sinuosity, as an extra latent vector that

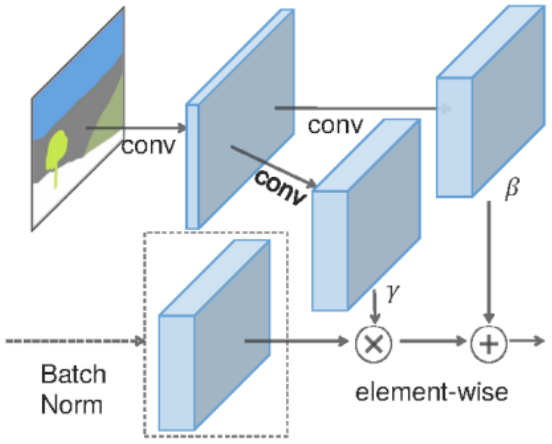


FIGURE 2.34: A schematic diagram of SPADE. From Park et al. [2019].

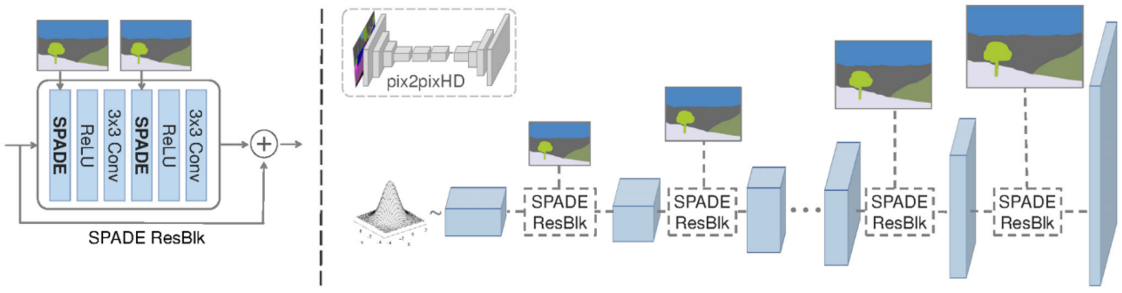


FIGURE 2.35: A schematic diagram of SPADE generator. From Park et al. [2019].

is concatenated to the latent noise vector as the input vector of the generator. In contrast, they used downsampling and convolutional layers to process the well data and soft data, called the channel complexity probability map, to create extra feature tensors for each convolutional network block. Their results proved the conditional GAN techniques condition GAN 2D realisations to the conditioning data from multiple sources (see Figure 2.36). However, one significant problem is that all conditional GAN techniques require training the conditional GAN for each specific case regarding the condition data. This means any pre-trained GAN is less likely to be reused if one or more conditioning data changes, which is very common in practice. For example, the entire model must be retrained to change the soft conditioning data from the channel complex probability map to the seismic amplitude map.

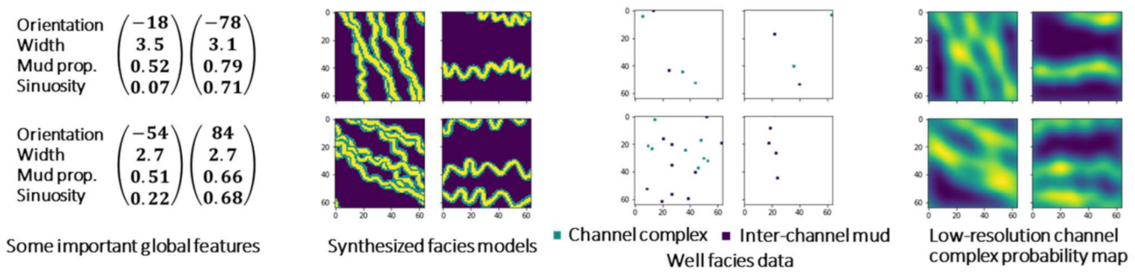


FIGURE 2.36: An example of Conditional GAN creating 2D realisations honouring multiple types of data. From Song et al. [2021a].

2.4 Summary of Chapter 2

This chapter overviewed three popular types of reservoir facies modelling tools: process-based models, geostatistical approaches and machine learning-based models. The process-based model involves geoscientists' understanding of how a reservoir formed in the sedimentary processes, whose realisations show highly plausible geological patterns close to the natural geology. Due to the computational cost and data conditioning difficulty, the process-based model suits to create conceptual geomodels used as the training data for geostatistical and machine learning algorithms. Geostatistics developed very well in the past decades, which is more flexible to condition realisations to diverse types of observed data, e.g. well and seismic data, than the process-based model. Still, geostatistics realisations are not as plausible as process-based realisations, particularly when the reservoir needs many facies to describe it. As both process-based and geostatistical models don't provide a direct parameterisation of their realisations (in terms of replacing simulated realisations with a small set of parameters), the modellers often need a dimension reduction tool to parameterise those simulated realisations, causing extra loss of geological realism. On the other hand, machine learning-based facies modelling tools, deep generative models, preserve high-level geological realism up to the same level as their training data and provide priors obeying either Gaussian or uniform distribution. However, the study of deep generative models is still at a starting stage that uses realisations from object-based models as the training dataset, leaving too many problems unsolved. For example, can deep generative models learn well from process-based models with multiple facies?

Based on the literature review above, I choose GAN to investigate further the potential of using deep learning-based algorithms to incorporate geological knowledge into geomodels. Compared to VAE, GANs' generations are less blurry in image synthesis and, therefore, show a clearer boundary between facies in geological modelling. This is important to facies modelling when there is no apparent ordering. To date of I started my PhD, and to the best scope of my knowledge, GANs succeeded in reproducing object-based models but fewer in learning from process-based models, especially for multiple (more than five) facies cases. Thus, I study to apply GAN to reproducing 3D multi-facies realisations from a process-based model, FLUMY, shown in the following chapters.

Chapter 3

Creation of Fluvial Training Dataset

Recent papers proved deep generative models' capability of reproducing the relatively complicated shapes of fluvial channel [Chan and Elsheikh, 2019, Laloy et al., 2017, 2018, Song et al., 2021b, Zhang et al., 2019b]. The training datasets in previous papers are simplified conceptual models of fluvial systems, mainly sinuous channels and associated levee. These datasets are not quite suitable to represent a meandering river, as they lack essential facies types, e.g. lateral accretion packages and various channel fills (please refer to the reviews in section 2.3.2 and 2.3.3). Usually, earlier studies used object-based simulators, e.g. TiGenerator [Maharaja, 2008] to create their training datasets, which are insufficient to represent the complicated placement of lateral and downstream evolution happening in natural fluvial rivers. On the other hand, facies simulations from process-based models largely preserve the plausible facies shapes and transitions by simulating sedimentary processes. Process-based models, therefore, are more suitable work as the training dataset to train deep generative models, but no pre-canned benchmark dataset in this level of complexity is available in the AI-based facies modelling application field. Though there are many process-based simulators, this study chose FLUMY [Grimaud et al., 2022] as it is fast enough to generate a number of models to cover the range of depositional parameter uncertainty. This chapter introduces the process-based training dataset created and used for this PhD study on applying GAN to 2D and 3D facies modelling.

3.1 Low NTG Meandering Simulations from A Process-based Model

3.1.1 FLUMY Simulation

MINES Paris Tech developed $FLUMY^{TM}$, a process-based model that builds facies realisations by a stochastic process of simulating channel migration and sediment deposition along the river in time sequence (Free download from `https://flumy.minesparis.psl.eu`) [Grimaud et al., 2022, Lopez, 2003]. This study uses version 5.9.12 of FLUMY software as the simulator to create various meandering fluvial simulations reflecting a range of sedimentary settings. FLUMY does not simulate the full physics of a river system, including dynamic turbulent flow motion, sediment transport and deposition, etc, to build up facies models. Instead, FLUMY sets up a couple of rules based on relationships between channel geometry, accommodation space, flow velocity, flooding and avulsion rates in fluvial systems to develop the model [Lopez et al., 2009]. FLUMY realisations are more plausible than object-based models because FLUMY simulates channel evolution, starting from initialising channel centreline, lateral migrating to develop meanders, and ending up with abandonment. By adjusting parameters, FLUMY can produce various scenarios with different channel sinuosity and sand-body patterns, achieving the favoured model effect. As a stochastic approach, FLUMY can create diverse facies realisations using the same settings but different random seed values to represent stochastic variability. So, FLUMY is a desired simulator to build a plausible meandering fluvial facies model set for GAN learning.

Facies models from FLUMY have nine facies that results from several geological processes, such as channel migration, avulsion, meander cut-off, levee breaches and aggradation (see Figure 3.1). By mimicking the evolution of meanders, FLUMY realisations capture the varying channel geometry and point bars accretion as sand deposits accrete on the channel's inner bank with the channel lateral migration. Depending on the downstream location, sand and mud deposits fill the channels, forming the sand and mud plug after channel abandonment occurs. FLUMY also captures the overbank and levee facies with the development of meanders, representing the occurrence of overbank flooding. As a product of levee breach occurrence, FLUMY uses a constant probability to add detailed crevasse splay sub-facies types, e.g. crevasse splay I, crevasse splay II and crevasse splay channel. Please, refer to section 2.1 for a detailed review of those geological processes.

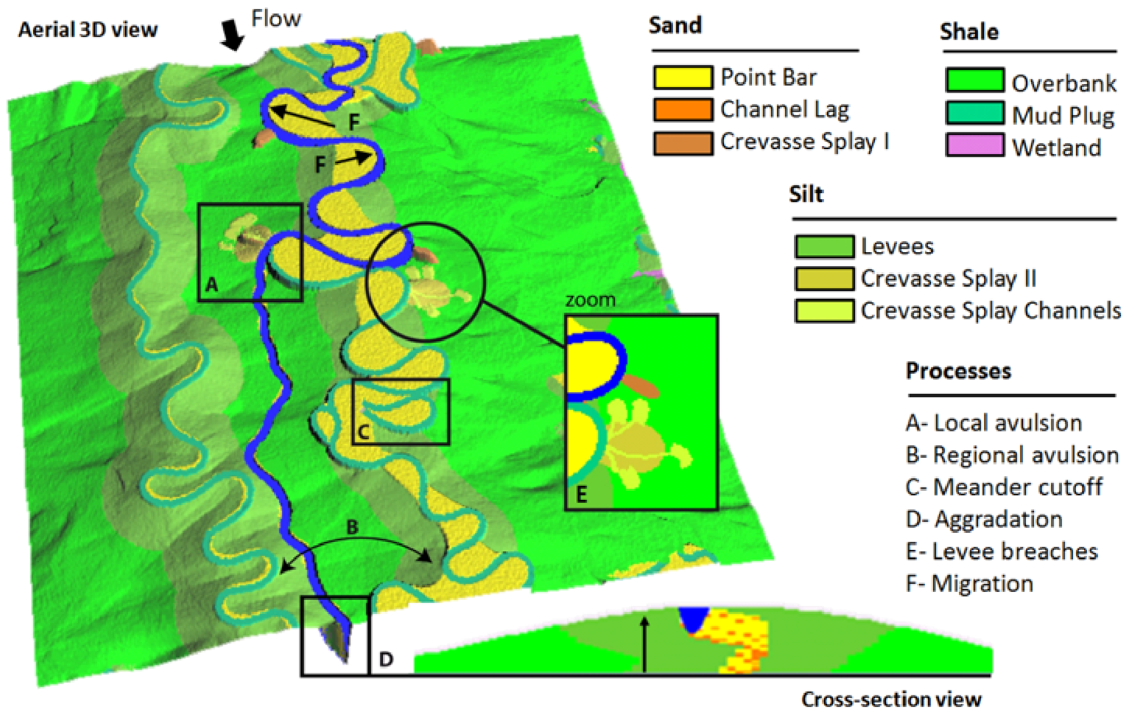


FIGURE 3.1: Processes of the meandering system in FLUMY. From $FLUMY^{TM}$ v.5.912 user guide.

3.1.2 Set Ups of Low NTG Meandering Fluvial Systems

This study focuses on low net-to-gross (NTG) meandering fluvial systems based on two considerations. First, low NTG meandering fluvial realisations have a bigger uncertainty of sand-body connectivity, which is a major consideration in current reservoir modelling pipelines because it further impacts most of the flow dynamic behaviour. For 3D models, the NTG threshold for connectivity is typically about 30% [Larue and Hovadik, 2006, Willems et al., 2017], so this study targets the NTG to around 20% to simulate a set of 3D models representing a wide range of connectivity. At the same time, the avulsion parameters do not significantly impact the net-to-gross of simulation outputs when the target NTG is 20%, and other parameters remain unchanged. Secondly, low NTG settings allow facies realisations to have less amalgamated channel elements, favouring more distinguished objects. This feature benefits the GAN performance evaluation in later chapters.

Determining the values of parameters in FLUMY relies on an embedded calculator, Nexus, designed for the non-expert user [Bubnova, 2018]. Each setting of FLUMY parameters runs five times to represent the natural system's aleatory uncertainty by the inherent stochasticity, e.g. the occurrence of overbank flooding and avulsion. Thanks to the stochastic process, FLUMY can simulate different realisations under the same sedimentary settings by varying the random seed values, which are '165426111', '165426222', '165426333', '165426444' and '165426555', respectively (see Table 3.1). The Nexus calculator has three inputs: channel maximum depth, sandbodies extension index (SEI) and net-to-gross (NTG). To control the variable, this study fixed the channel maximum depth to 5 meters in Nexus to automatically calculate the rest channel geometry parameters in FLUMY. Based on a trial and error, FLUMY outputs have a net-to-gross of about 20% when the net-to-gross parameter in Nexus equals 10%. By varying the SEI parameter, the Nexus calculator can auto-fill the values of different parameters related to avulsion in FLUMY. To verify the auto-filled avulsion period range, the records of the modern river from Slingerland and Smith [2004] work as a reference in this study. Five values of the SEI used in this study are the minimum (20), the maximum (160) and three unique values marking three different sand-body scenarios: ribbon type (50), standard type (80) and sheet type (110). Table 3.1 summarises the necessary parameters in Nexus used to create the 25 3D simulations. Nexus automatically calculates the rest of the FLUMY parameters. Except for the avulsion parameters, the FLUMY parameters are the same in all 25 simulations (see Table 3.2).

TABLE 3.1: FLUMY parameters in Nexus to create 3D simulations

Parameters	Values
Channel Maximum Depth	5
Sandbodies Extension Index	20,50,80,110,160
Net to Gross (%)	10
Grid Lags (DX, DY)	10
Give Number of Nodes (NX, NY)	256
Seed	165426111,165426222,165426333, 165426444,165426555

The setting above yields a set of low NTG (around 20% ranging from 10% to 23%) facies models with different avulsion rates in a moderate sedimentation rate (about 0.4 cm/year)

Table 3.2: FLUMY parameters determined by Nexus

Parameters	Values
Erodibility Coefficient	5.16×10^{-8}
Channel Width, Wavelength, Variable Scale (m)	50, 625, 100
Domain Margin (Multiple of Channel Width)	12
Flow Direction (Degrees Clockwise)	90
Slope Along Flow	0.001
Levee breaches during Overbank Flow	No
Probability for Transition from CSI to CSII	0.5
Probability for Adding a New CS Channel	0.9
Equilibrium Profile Elevation, Changes (m)	9999, Never
Aggradation Type, Occurrence	Overbank Flow, Poisson(100it)
Aggradation Thickness, Exponential Decrease (m)	Normal(0.167,0.05), 1532
Levee Width (Multiple of Channel Width)	6
Wetland Proportion (%)	0
Draping Intensity (m/10000it)	0

environment. This dataset shows a range of plausible meandering fluvial patterns, covering a range of uncertainty (see Figure 3.2). Frequent avulsions result in the river having a shorter time to develop meanders and favouring more ribbon-type sandbodies in vertical sections and a smaller channel sinuosity in plane view (Figure 3.2a). On the other hand, a lower avulsion rate gives the channel more time to develop, leading to more sheet-type sandbodies in vertical sections and a bigger channel sinuosity raising meander cut-offs in plane view (Figure 3.2c). Based on modern rivers' records, Slingerland and Smith [2004] suggested the avulsion period varied between 28 to 1400 years. While they also mentioned this range is not a theoretical limit to a realistic choice of avulsion period because a smaller or bigger avulsion period may exist [Slingerland and Smith, 2004]. Therefore, the above range is extra evidence for justifying the selected avulsion periods in this study. Table 3.3 summarises the relationship between sandbodies extension index values and the corresponding avulsion parameters in FLUMY used for the five groups' settings.

Table 3.3: Avulsion parameters used in FLUMY

Group Number of different avulsion parameters	1	2	3	4	5
Sandbodies Extension Index	20	50	80	110	160
Regional Avulsion Period, T_R	190	480	800	1000	1500
Local Avulsion Period, T_L	105	270	435	600	885
Total Avulsion Period	68	173	282	375	557

Avulsion includes regional avulsion and local avulsion; $\frac{1}{AvulsionPeriod} = \frac{1}{T_R} + \frac{1}{T_L}$

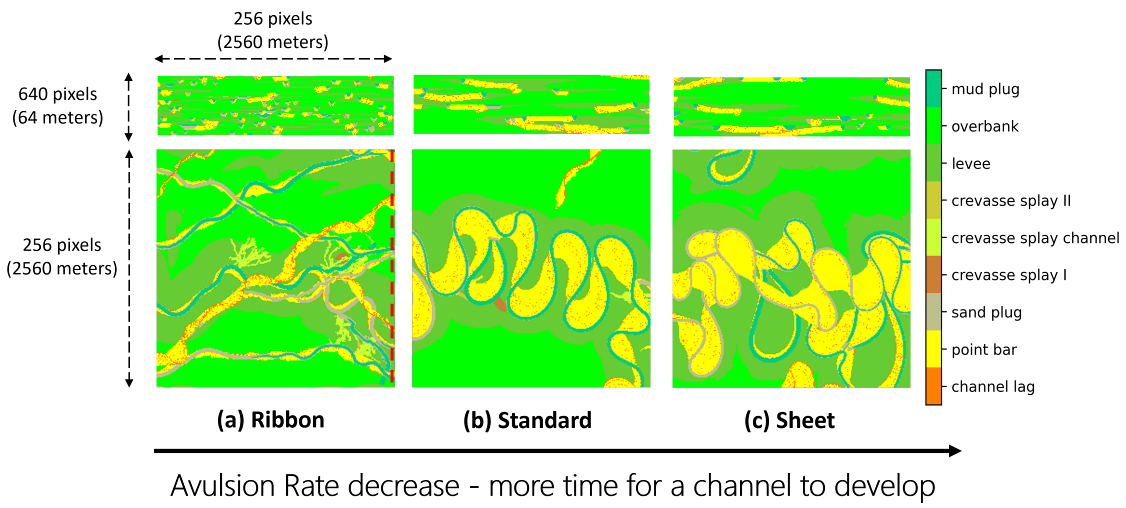


FIGURE 3.2: FLUMY 2D realisation with different avulsion rates. Vertical sections are picked along the normal direction of the global slope at the downstream side (red dash line) and are ten times exaggerated in vertical size. (a) A facies model produced by a high avulsion rate setting. (b) A facies model produced by a moderate avulsion rate setting. (c) A facies model produced by a low avulsion rate setting.

3.2 GAN River-I Training Dataset Production

3.2.1 Three Datasets Representing Different Levels of Data Complexity

Using pre-canned training datasets to train deep generative models can reduce computation costs in speed and storage in the data reading and pre-processing. Though the 3D simulations from FLUMY can directly serve as the feedstock for GANs, the training program needs to pre-process the data before feeding it into the GAN model for every iteration. Particularly for this study case, the FLUMY 3D simulation is a big (in terms of the number of voxels) training data and, therefore, often needs data pre-processing, e.g. random cropping, value transformation, etc., which takes time for the CPU to read and process the requests.

For the convenience of GAN training, an extra program of making the dataset slice all FLUMY 3D facies realisations every 0.1 meters vertically to transform and store the training data in the format of 2D data/images with a clear name labelling the sample and slice. To avoid undefined values, the program only picked the section between 0 to 64 meters in all 25 FLUMY simulations and sampled them every 0.1 meters. Therefore, the program converts each 3D FLUMY simulation into 640 2D data/images whose size is 256 × 256, and the training dataset contains 16,000 data/images in total. This training dataset is the 9-Facies dataset in GAN River-I. To make this training dataset reusable in the 3D GAN study, each 2D training data has a name: 'Group_Sample_Layer_Facies.format'. For example, a 2D data named '2_1_0_Facies.npy' indicates it is the base (0th) slice of the first sample in the second avulsion rate group and stored in Ndarray format.

For the demand of developing deep generative models to learn complex patterns started from simpler cases, another program created the 7-Facies dataset and 3-Facies dataset by grouping the 9-Facies dataset. The main consideration of the reduction is the sedimentary relationships between different facies and their likely rock properties, e.g. permeability, representing subsurface reservoirs' flow units. The 7-Facies dataset uses a 'crevasse splay' facies to represent three facies in the 9-Facies dataset: 'crevasse splay I', 'crevasse splay channel' and 'crevasse splay II', and keeps the rest of facies unchanged. Then, each facies variable uses one integer as its unique code in the training dataset. The program used a similar process to create the 3-Facies dataset but a different grouping rule. The 3-Facies dataset replaced the 'channel lag' and 'point bar' facies in the 9-Facies dataset with the new 'point bar' facies since both are part of the lateral accretion packages in the meandering fluvial system. A new 'channel' facies in the 3-Facies dataset replaced the 'sand plug' and 'mud plug' facies in the 9-Facies dataset because they are both channel-fills in abandoned channels, though their rock properties are significantly different, e.g. porosity and permeability. Figure 3.3 presents an example of the same data in three different cases.

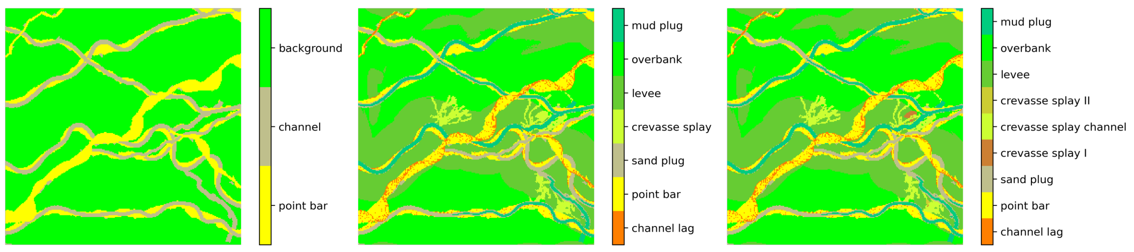


FIGURE 3.3: An example of FLUMY data in the three training datasets.

The three training datasets in GAN River-I provide different levels of data complexity in the number of facies, catering for the demand of testing GANs in learning meandering fluvial systems with multiple levels of detail. All facies have unique codes in these training datasets for the convenience of using deep generative models that usually only tackle numerical data (see Table 3.4).

TABLE 3.4: Relationship between $FLUMY^{TM}$ Facies and Dataset Codes

Facies	9-Facies	7-Facies	3-Facies
Channel Lag	1	0	0
Point Bar	2	1	0
Sand Plug	3	2	1
Crevasse Splay I	4	3	2
Crevasse Splay II Channel	5	3	2
Crevasse Splay II	6	3	2
Levee	7	4	2
Overbank	8	5	2
Mud Plug	9	6	1

The 9-Facies dataset keeps the original facies code in FLUMY, preserving all detailed facies resulting from modelled geological processes. Channel lag (coarse residual) facies and point bar (sand) facies compose lateral accretion packages at the channel inner bank with lateral migration. Channel abandonment, e.g. avulsion and meander cut-off, results in the sand (sand plug) and mud (mud plug) deposit in abandoned channels. Sediments flow out of the channel and deposit from proximal to distal, transitioning from silt to shale on the flooding plain, forming levee facies and overbank facies every time overbank flooding occurs. Once the levee breaches, water takes erosive deposits and places aside the meander forming the Crevasse-splay-I facies. A constant probability controls the change of erosive sediment to non-erosive, making another facies, Crevasse-splay-II. Another constant probability decides whether to add a channel to the non-erosive deposits, creating the Crevasse-splay-Channels facies. As this training dataset has all facies transitions no matter how often, it is suitable to test GANs' learning capability of multi-facies distribution resulting from detailed geological processes. However, this training dataset involves too detailed crevasse splay descriptions with a tiny proportion, which current deep generative models easily ignore. Therefore, this study discards using this dataset and leaves the challenge of fully capturing all processes' mechanisms to future studies.

The 7-Facies dataset simplifies the original facies models by grouping the crevasse splay-associated facies while preserving facies with contrasting rock properties. Of course, crevasse splay deposits may take up a significant proportion of the fluvial facies models and contain complex facies distribution [Burns et al., 2017]. However, this is not the case in the FLUMY simulations with the settings in this study, as crevasse splay deposits take about 0.4% in total. Still, the 7-Facies dataset preserves detailed facies distribution in fluvial systems, impacting the flow response of subsurface reservoirs. This dataset can also work as an ensemble of geomodels with complex sand distribution, describing the connected point-bars scenario shown in Donselaar and Overeem [2008]. This dataset is the benchmark for testing advanced deep generative models in reproducing multi-facies distribution and is therefore used as the training dataset for the rest of multi-facies 2D and 3D modelling studies in Chapter 4 and 5.

The 3-Facies dataset, on the other hand, narrows the scope to channel geometry and lateral accretion package placement. So, this dataset suits to test how much deep generative models can learn to reproduce geologically plausible shapes. Especially this dataset can work as a benchmark for a comparison study of different GAN models because three facies is almost the biggest number of facies used in recent works [Song et al., 2021b, Zhang et al., 2019b]. For example, this thesis compares different GANs' performance on learning meandering channels in Chapter 4 using this dataset as the training dataset. As this dataset doesn't contain detailed rock types in the channel, it can work as an ensemble of geomodels, describing the isolated point bars scenario shown in Donselaar and Overeem [2008].

3.2.2 Quantitative Analysis of GAN River-I

This section introduces quantitative analysis tools used to measure the training dataset as a reference for evaluating GANs' performance. Indeed, the quantitative evaluation of GAN is a hot and open question in the GAN application research field, which is still controversial [Borji, 2019]. This study only uses classical quantitative scores, including net-to-gross, connectivity, etc., which will be described below, to capture the geological realism at a certain level instead of thoroughly investigating how to quantify the geological realism of the GAN River-I dataset and GAN results.

3.2.2.1 Global Indicators

Global indicators referring to the entire model/deposit, e.g. facies proportion and connectivity, describe the 3D models statistically using simple scores with low computation costs. Those indicators measure simple but important global features that are often the first considerations in subsurface modelling.

Facies proportion describes the area/volume of certain facies in the modelled domain [Rongier et al., 2016]. This study uses equation 3.1 to approximate the facies proportion and takes the sand-prone facies proportion as the NTG.

$$Facies\,Proportion = \frac{N_{facies}}{N_{total}} \tag{3.1}$$

where N_{facies} is the total number of cells of the facies; N_{total} is the total number of cells. For the GAN studies for multi-facies modelling, three facies in the 7-Facies dataset are sand-prone (net) facies for the NTG and connectivity calculations: channel lag, point bar and sand plug. This grouping slightly underestimates the actual NTG and sand connectivity because it ignores sand (net) facies from other sources, e.g. crevasse splay. Still, the NTG and connectivity calculated this way adequately evaluate GANs' performance as this grouping applied to both the training data and GANs realisations.

Connectivity characterises how well the facies are connected in the discretised grids based on the connected component, meaning the connected cells belong to the same connected body, which often refers to static connectivity in reservoir studies [Hovadik and Larue, 2007, King, 1990, Rongier et al., 2016]. A cell has three possible neighbourhoods depending on the connection type (see Figure 3.4). This study uses the Cartesian grid, thus, taking the face-connected definition of the connected neighbourhoods. As for the connectivity calculation, the biggest connected sand-body connectivity and the facies connected probability are two widely used connectivity indicators in literature [Hovadik and Larue, 2007, King, 1990].

The biggest connected sand-body connectivity is defined as the area/volume ratio of the biggest connected component to the total components of the facies [King, 1990], given by

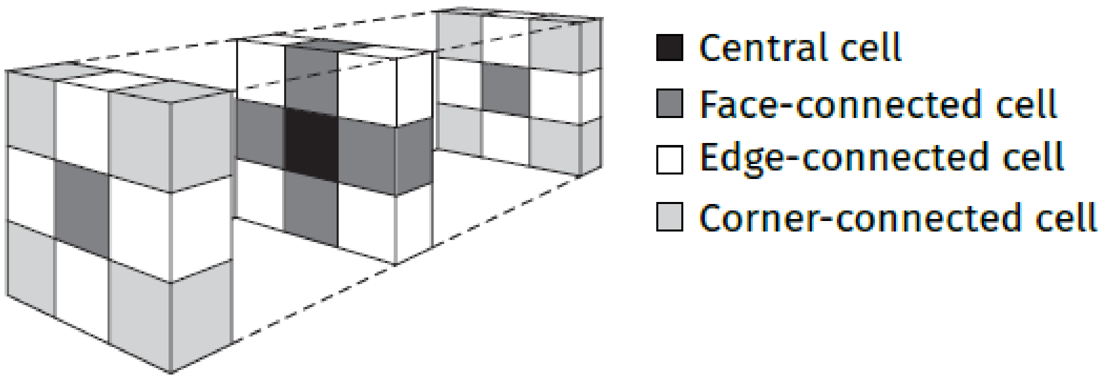


FIGURE 3.4: Different types of connected neighbourhoods for a given central cell. From Deutsch [1998], Rongier et al. [2016]

Equation 3.2

$$C_{biggest} = \frac{n^{max}_{facies}}{N_{facies}} \tag{3.2}$$

where n^{max}_{facies} is the number of cells of the largest connected component of the facies; N_{facies} is the total number of cells of the facies.

The facies connected probability refers to the probability of two cells with the same facies belonging to the same geo-body [Hovadik and Larue, 2007], given by Equation 3.3

$$C_{probability} = \frac{1}{N^2_{facies}} \sum_{i=1}^{n_{facies}} (n^i_{facies})^2 \tag{3.3}$$

where N_{facies} is the total number of cells of the facies; n_{facies} is the number of connected components of the facies; n^i_{facies} is the number of cells of the connected component i associated to the facies.

This study uses the NTG and the biggest connected sand-body connectivity to illustrate that the GAN River-I dataset covers a wide range of connectivity within a similar NTG. To calculate the area/volume of the connected bodies, this study uses a Python package, scikit-image, to pick the cells belonging to the same connected body [van der Walt et al., 2014]. Due to the limited number of 3D models available, four sets of 3D cubes are created by sampling the original 3D models with a stride of 1 and thickness of 64, 32, 16, and 8, respectively. (See Figure 3.5). As mentioned earlier, the calculation operates on sand-prone facies, including channel lag, point bar and sand plug, and treats the crevasse splay-associated facies as silt for simplicity, though the crevasse splay I and crevasse splay channel are often sand-prone. Those four connectivity plots show the data point spread widely in the connectivity domain within an NTG of mostly about 0.18. When sampled with a smaller thickness, the cubes cover wider areas in the NTG-connectivity 2D domain. The later NTG-connectivity plots for 2D models can be regarded as a product of subsampling with a stride of 1 and a thickness of 1 slice (Figure 3.7).

Based on percolation theory, researchers used S-curves to fit the relationship between NTG and connectivity, describing the reservoir connectivity quantitatively [King, 1990, Larue and Hovadik, 2006]. The connectivity rapidly increases in the S-curve plot once the NTG surpasses the percolation threshold, around 0.3 for the 3D models (see Figure 3.6 a) [King, 1990, Larue and Hovadik, 2006]. However, geological features may cause connectivity for a given proportion to increase or decrease, resulting in a divergence between the percolation threshold in theory and natural geological systems. For example, channelised sand with a more meandering shape will lead to greater overall connectivity than those with straight channels, which need a bigger NTG to reach the same volume of connected sand-bodies [Larue and Hovadik, 2006]. Individual realisations in an ensemble of stochastic geological models would scatter within an area around an S-curve called the cascade zone. The ensemble's average response may shift to the left or right side of the S-curve, indicating the geological features increase or decrease the sand-body connectivity. So, for individual realisations, their relationships of NTG and connectivity might differ from the predicted values by the S-curve based on the percolation theory alone.

To avoid the repetition of 2D facies slices, fifty 3D facies realisations with a 32-meter thickness compose a reference set for the sand connectivity and proportion in 3D. This reference set is originally from GAN River-I, which has twenty-five 3D FLUMY models with 64 meters of thickness. The reason for creating this reference set is the 3D study in Chapter 5 tests the algorithm to build 3D facies models with 32 meters for a lower cost of storing simulated ensemble. Splitting those FLUMY models in the middle makes them a fifty-point set, yielding the sand connectivity plot against NTG (see Figure 3.6 b). The NTG of individual models in the 3D reference set is around 0.2, smaller than the theoretical threshold value (0.3) of the percolation model in the 3D case because this reference set origins from a low NTG meandering fluvial set, GAN River-I.

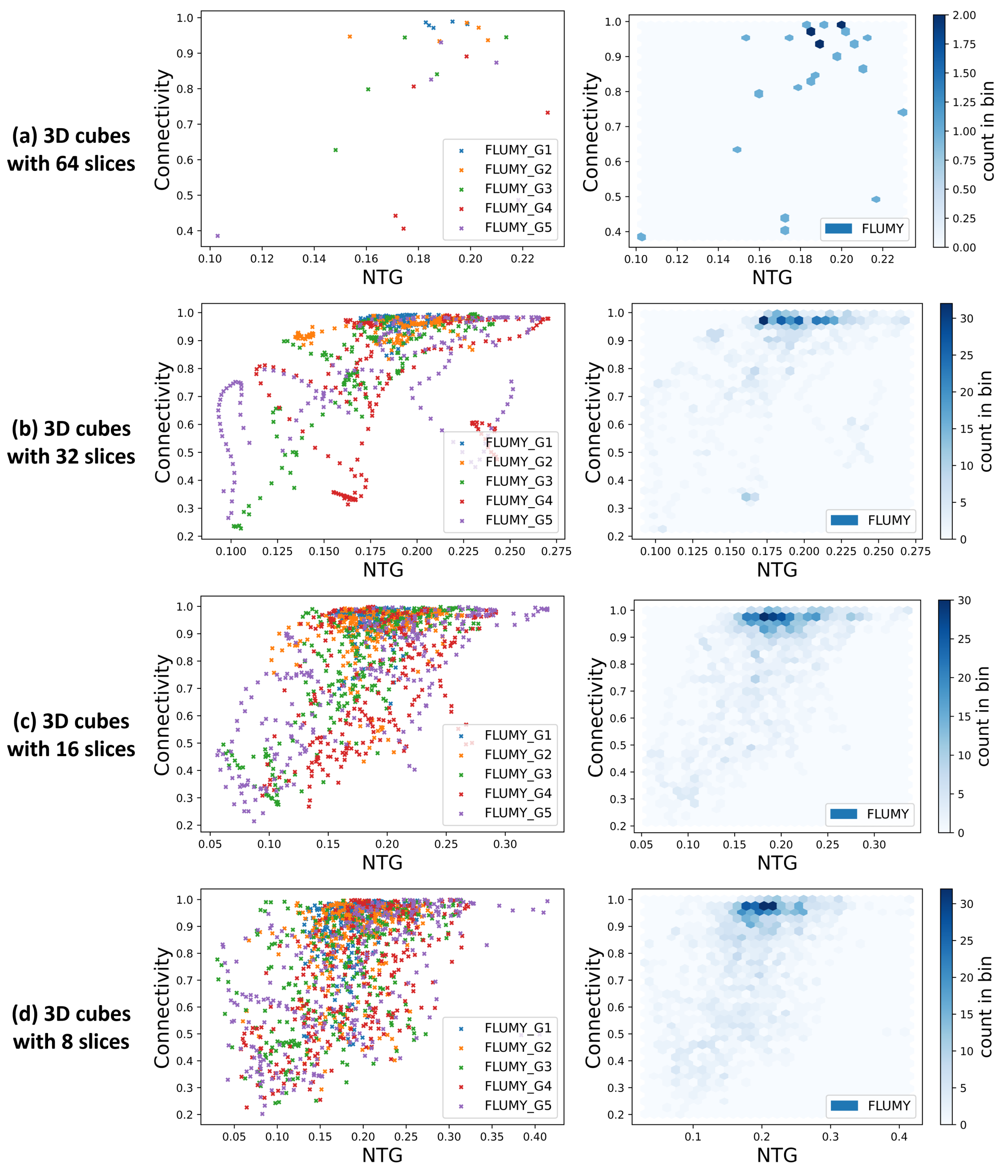
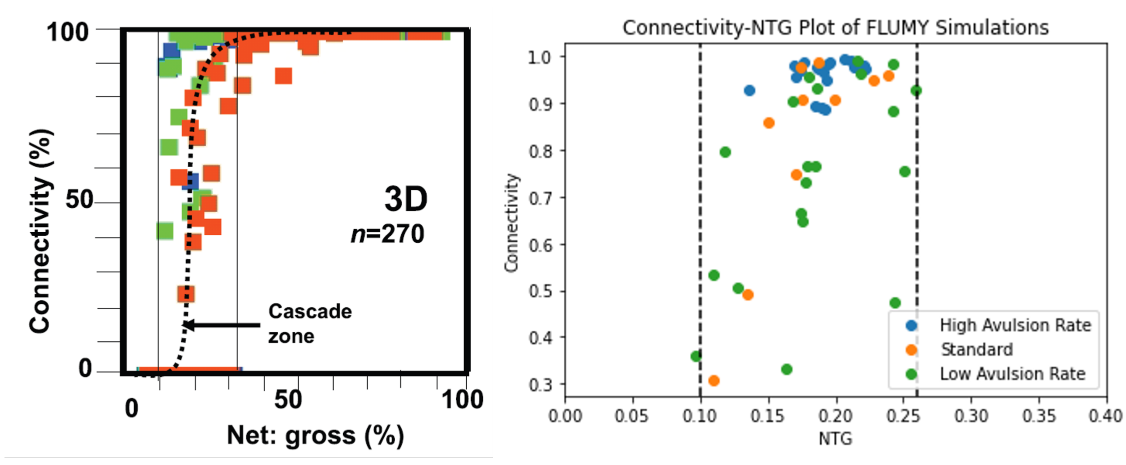


FIGURE 3.5: Plots of Connectivity VS NTG for the sampled 3D cubes from the 25 FLUMY 3D facies models. FLUMY_G1 to FLUMY_G5 in the left column refer to the avulsion rate groups in Table 3.3, denoting avulsion rate from high to low (avulsion occurrence period from short to long). Plots in the right column are corresponding Hexbin plots of all 3D cubes. The 3D cubes are the results of sampling with the stride of 1 and the thickness of (a) 64, (b) 32, (c) 16, and (d) 8 slices, respectively.

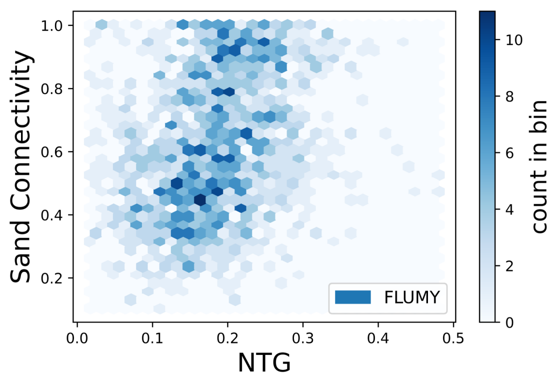


(a) A diagram of percolation theory **(b) Connectivity VS NTG plot of FLUMY simulations**

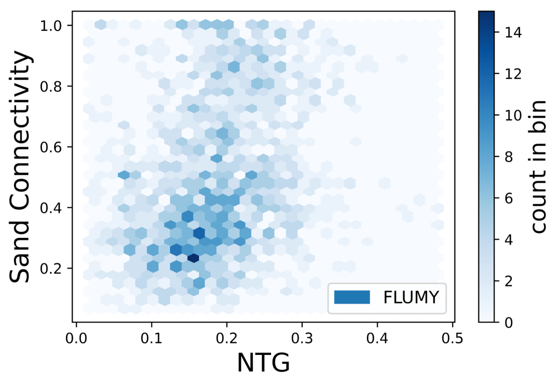
FIGURE 3.6: Plots of sand connectivity in 3D against proportion. (a) A schematic diagram of percolation theory, showing the S-curve and Cascade zone. Modified after Larue and Hovadik [2006]. (b) A plot of sand connectivity against proportion for 50 FLUMY simulations with 32 meters of thickness. The area within the two black dashed lines is the Cascade zone.

As Figure 3.6 b) shows, this fifty-point dataset's scatter greatly diverges from the idealised S-Curve prediction and gets high connectivity at a lower NTG threshold value at high avulsion and, therefore, lower sinuosity realisations. This is because of the joint effects of channel developments and avulsions. The high avulsion rate simulations (groups 1 and 2) generally have high sand-body connectivity as the frequent change of channels increases the probability of connecting sand bodies in 3D. While sand in the low avulsion rate simulations (groups 4 and 5) more likely distributes laterally, forming sheet-type point bars, occasionally avulsion changes the sand's lateral accretion place, which may result in isolated point bars. Those models have low NTG and moderate sedimentation rates, meaning that mud takes a much bigger proportion than sand and deposits at a reasonable speed, increasing the chance of sealing the old sand bars when the meander takes a long time to develop at other locations. Thus, the low avulsion rate simulations show a bigger variation in connectivity. The high scatter in connectivity for a given NTG suggests that GAN should honour the high uncertainty in the predicted connectivity of this type of geology in its realisations.

The same analysis is also carried out on the 2D slices as the later study in Chapter 4 aims to reproduce 2D models first. Instead of plotting all data, a subset samples 10% from GAN River-I with 16,000 2D slices by picking every ten 2D slices, which ensures a good diversity in a dataset volume of 1,600 FLUMY realisations. Due to the nature of the fluvial 3D model, any 2D layer resembles layers immediately above or below it because river channels have a few meters in depth, while the interval between image slices in GAN River-I is only 0.1 meters. Unlike the connectivity of 3D models, both low and high avulsion rates 2D facies models show a wide range of connectivity and overlap among different groups as the 2D models have bigger percolation threshold of connectivity than the 3D models [Masihi and King, 2012]. Therefore, this study plots all 2D models together instead of further categorising them based on the avulsion rates in the connectivity plot (see Figure 3.7). Data points spread wide in the two connectivity indicators and NTG space, which can be used as the reference later to analyse the diversity of GANs' generations.



(a) The biggest connected sand-body connectivity



(b) The facies connected probability connectivity

FIGURE 3.7: Plots of Connectivity VS NTG for the subset 2D facies models.

Another connectivity plot calculates the probability of two cells with the same facies belonging to the connected component against the lag distance [Renard and Allard, 2013]. This study uses a Python library, loopUI-0.1, to plot the sand connectivity probability curves for all 2D slices, which calculates the connectivity probability along the lag distance by Equation 3.4 [Pirot et al., 2022]

$$\tau(h) = Prob(Connected(s, s+h) \mid X(s) = 1, X(s+h) = 1) \tag{3.4}$$

where h denotes the lag distance, s denotes the start point, $Connected$ refers to the function of judging if the two cells are connected based on the definition of the face-connected neighbourhood in Figure 3.4, and X is the facies variable of the given cell. This plot illustrates GAN River-I dataset has a wide range of connectivity probability (see Figure

3.8). The red solid line shows the mean value, and the black box plot indicates 75%, medium, and 25% cases of the GAN River-I dataset.

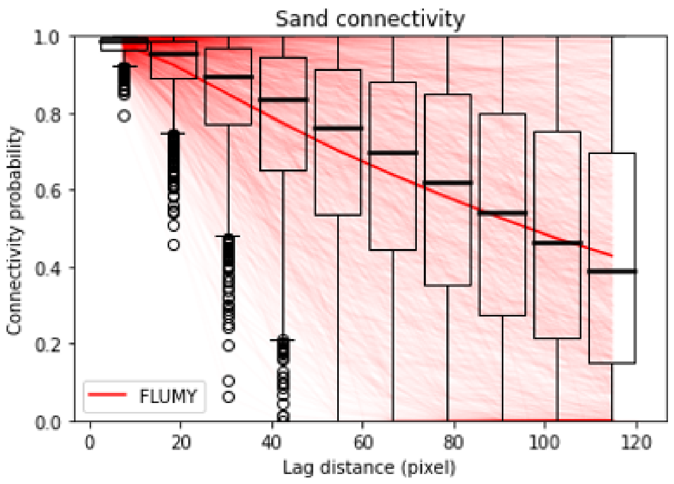


FIGURE 3.8: Plots of sand connectivity probability as a function of lag distance for the subset 2D facies models. Each line represents one sample's sand connectivity probability.

3.2.2.2 Low Dimensional Representation Visualisation

Assessing model diversity in GAN River-I provides a reference for the mode collapse detection, which is an important indicator of evaluating GANs' performance in later chapters. Analysing the similarity in a dataset often relies on dimensionality reduction tools, such as t-SNE and UMAP, to visualise high dimensional data in low dimensional space (2D or 3D) [McInnes et al., 2018, Van der Maaten and Hinton, 2008]. This type of analysis interprets the model diversity in an ensemble of data based on their global structure and the local structure, providing a richer information-involved representation than classical global indicators, for example, sand proportion and connectivity. A plot of the training dataset can work as the reference to check if GANs learn diverse types of data, which is particularly important to detect the mode collapse issue mentioned in section 2.3.3.

This study employs a dimension reduction technique that allows visualising the ensemble of models' diversity, UMAP, to assess the diversity of the GAN River-I dataset by projecting it into a 2D metric space. UMAP assumes high dimensional data has a uniform distribution on a locally connected manifold and uses a k-neighbour-based graph learning to estimate this local manifold to represent the data in a low dimension [McInnes et al., 2018]. Compared to other popular dimension reduction algorithms, e.g. t-SNE, UMAP runs faster, which is why this study chose UMAP to visualise GAN River-I. UMAP's random number generator allows using random seeds to repeat the transformed results. This study fixed the number of neighbourhoods, $n\_neighbors$, as 15 and the effective minimum distance between embedded points, min_dist, as 0.1; both are UMAP parameters to impact the estimation result of the local manifold. A big $n\_neighbors$ makes the manifold involve more global features, while the manifold considers more local features when the number of the neighbourhood is small. In contrast, the min_dist determines the appearance of UMAP results; a smaller min_dist leads to more clumped embedded points, while a larger min_dist makes the embedded points spread wider.

One analysis interprets the distribution of GAN River-I data from different avulsion rate groups in UMAP space. Figure 3.9 shows the data spread widely in the UMAP space, and the data clouds from different avulsion rate groups overlap significantly. Primarily, the high avulsion rates (Group 1 and 2) data spreads closer to each other and largely overlaps data clouds from other groups. With the decrease in avulsion rate (group changes from 1 to 5), the data points gradually spread wider and outer to the centre area of the embedded cloud. The low avulsion rates (Group 4 and 5) data spreads much broader, and many distribute at the outer edge of the embedded plot of GAN River-I. As later GAN studies in this thesis use the same UMAP projector, this interpretation provides a basic insight into the geological patterns' diversity, which can be used as the reference to detect mode collapse and check which type of geological patterns are missed.

Another analysis verifies the correlation between the meandering pattern and the coordinates in UMAP space by picking data from the central and outer areas of the data cloud. Five points, including the minimum or maximum values in each dimension and one point in the middle of the cluster, highlight their geological patterns in the original data space (see Figure 3.10). The data in the middle of the UMAP cluster show a clear thin meander belt with less extended sand bodies. In contrast, the four points at the outer area all represent highly meandering models with wide extended sand bodies. This observation consists of the plot of data against avulsion rates. FLUMY uses the same initialisation of channels when given the same random seed and creates new thin channels when avulsion occurs. A moderate sedimentation rate allows a slight slope of sand bodies in the vertical section. After the discretisation with very small intervals (0.1m), some horizontal slices

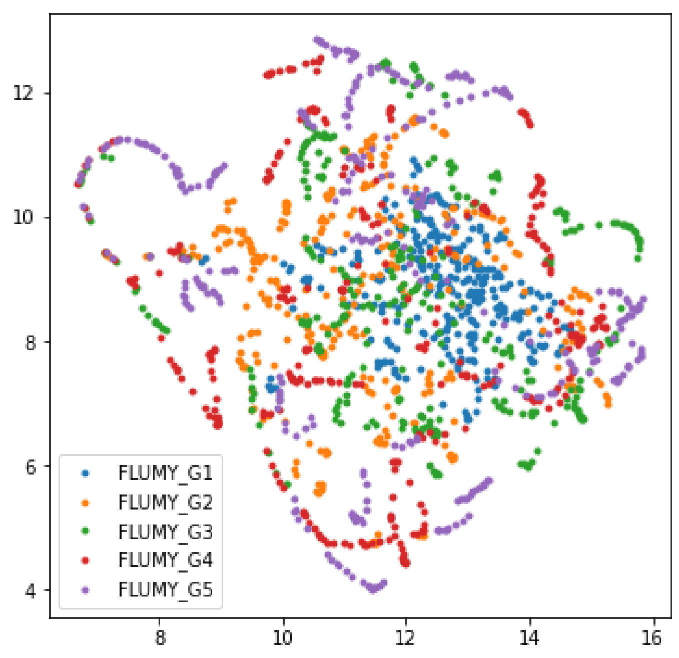


FIGURE 3.9: Plot of the GAN River-I dataset in the UMAP 2D space. FLUMY_G1 to FLUMY_G5 denote FLUMY realisations from groups 1 to 5 in Table 3.3, denoting avulsion rate from high to low (avulsion occurrence period from short to long).

contain partial sand bodies, either the bottom or the top of the meander belt, presetting thin meandering patterns. Therefore, the narrow meandering patterns also exist in the low avulsion rate training images, accounting for the overlap in the UMAP space.

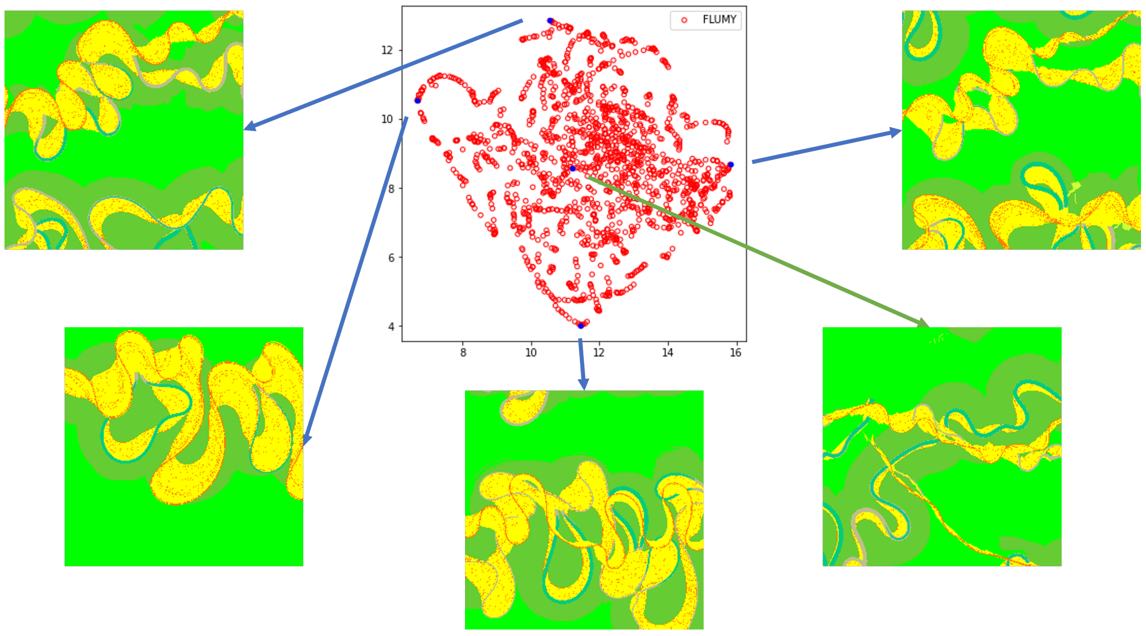


FIGURE 3.10: Five points interpretation of the UMAP 2D space.

A further study traces the sequential slices of a depositional succession in UMAP space to study the relationship between FLUMY depositional patterns in data and UMAP metric space. This analysis uses sequential slices from a given succession containing a channel evolution in 3D. Another 3D succession would be in a different location in the UMAP space due to stochasticity. Figure 3.11 shows the selected succession in the UMAP metric space. With the lateral development of the point bar, data highlighted in blue locates closer to the data cloud's edge in UMAP metric space. This trend continues till the channel abandonment occurs leaving a narrow residual upper part and a new narrow channel emerges in the simulated area. This study correlates the channel evolution to the coordinates change in UMAP metric space, accounting for high avulsion rates FLUMY models from Group 1 and Group 2 cluster centrally but FLUMY models with lower avulsion rates (Group 3, 4, 5) distribute wider in the UMAP visualisation plot. A low avulsion rate allows channels to have a longer time to develop laterally than a high avulsion rate.

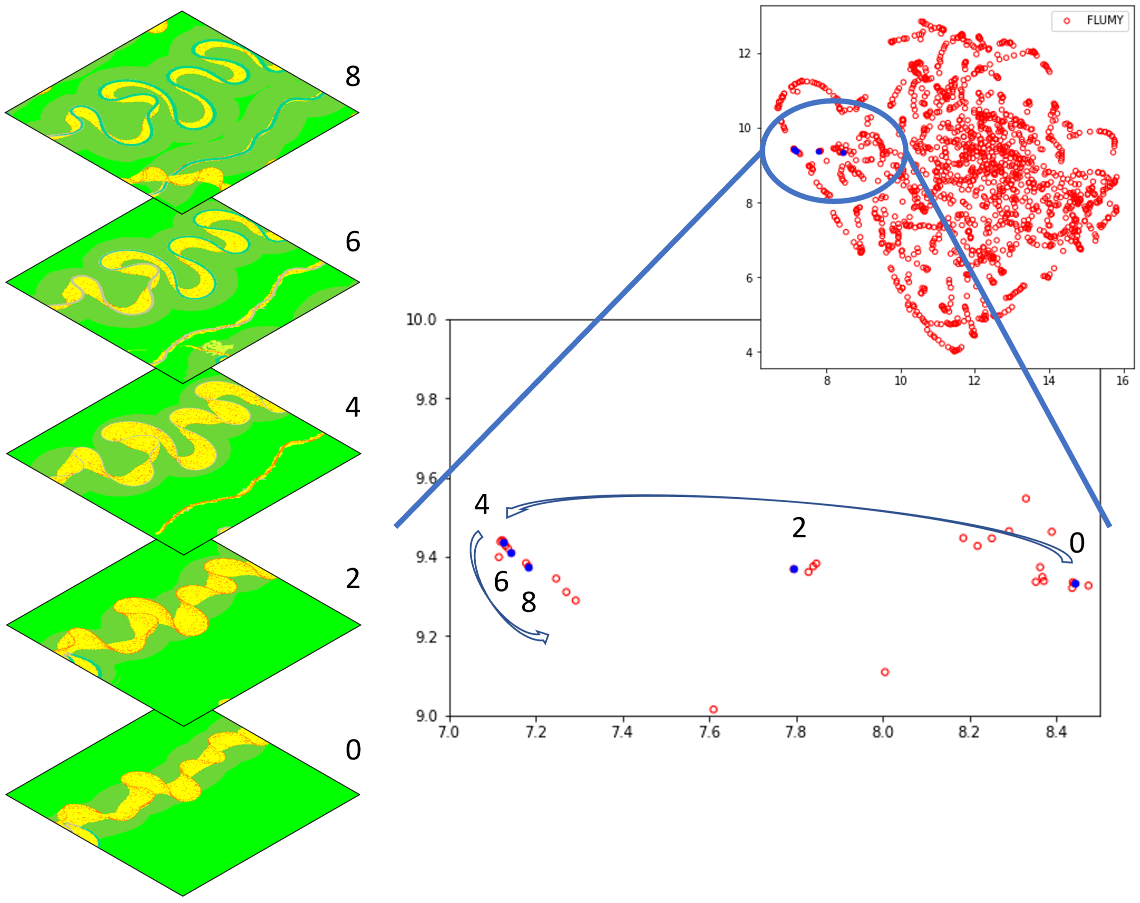


FIGURE 3.11: A vertical set of FLUMY realisations in UMAP 2D space.

3.2.3 Qualitative Analysis of GAN River-I and A Proposed Qualitative Score

Qualitative analysis of geological realism often relies on visual interpretation based on human knowledge and understanding of geology. Though subjective, human interpretation provides a thoughtful judgement on the agreement between modelled and natural geology that is hard to quantify fully. GAN River-I is a set of 2D slices according to elevation instead of the depositional surface. A 2D slice may contain detached geobodies or debris, while the slice above or below may contain additional geometric information that shows the detached geo-body or debris trace is part of a meander. However, those unfavourable features introduce extra learning and evaluating difficulties because geologists might also label those images as bad quality results. As mentioned in section 3.1.1, FLUMY is built upon a set of assumptions and is not designed to capture all detailed features of a realistic meandering fluvial reservoir, e.g. shale drapes between point bars. This project aims to reproduce FLUMY realisations with its strengths as well as flaws and refers to the facies models from FLUMY as the 'ground truth'. Because this project aims to recreate the complex outputs of the FLUMY models using GANs, not to replicate a natural river system. The best result of machine learning-based models, such as GANs, is achieving comparable model quality as its training dataset.

This study highlights two types of 'artefacts' caused by discretisation in GAN River-I. The debris traces make the training data 'noisy', containing dot points or dash lines (see Figure 3.12). Considering the channel lag facies (orange colour) often show dot points/dash lines features, the debris traces might introduce more bias on this type of feature but should not ruin GAN generations' quality because it only takes up around 0.5‰ in GAN River-I. In contrast, the detached geo-body exists in a few 2D slices, taking up about 1‰, which is part of a meander belt above or below it (see Figure 3.13). Those detached geobodies may encourage GANs to create artefacts when using GAN River-I to train GANs to reproduce 2D slices.

Inspired by the qualitative analysis of GAN River-I, a proposed occurrence-based score

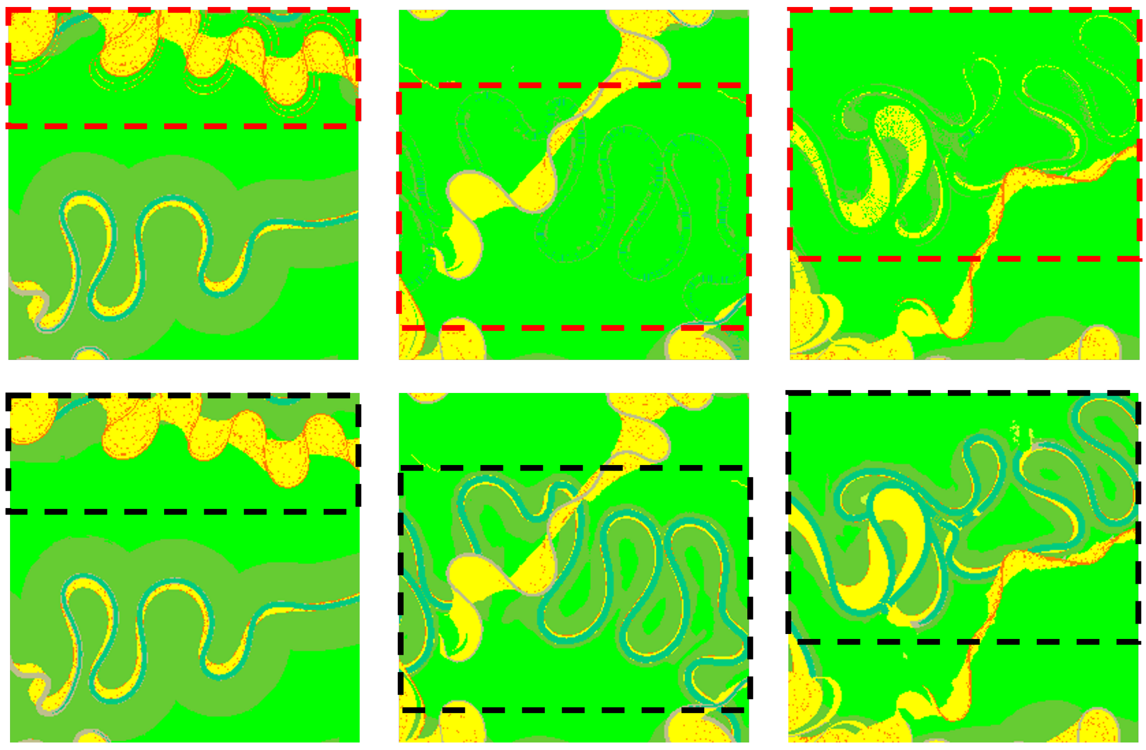


FIGURE 3.12: FLUMY 2D slices containing debris trace. The red dashed rectangular line highlights the location of the debris trace. The black dash line rectangular shows the corresponding geo-body above or below the realisations with debris trace (red dash line rectangular).

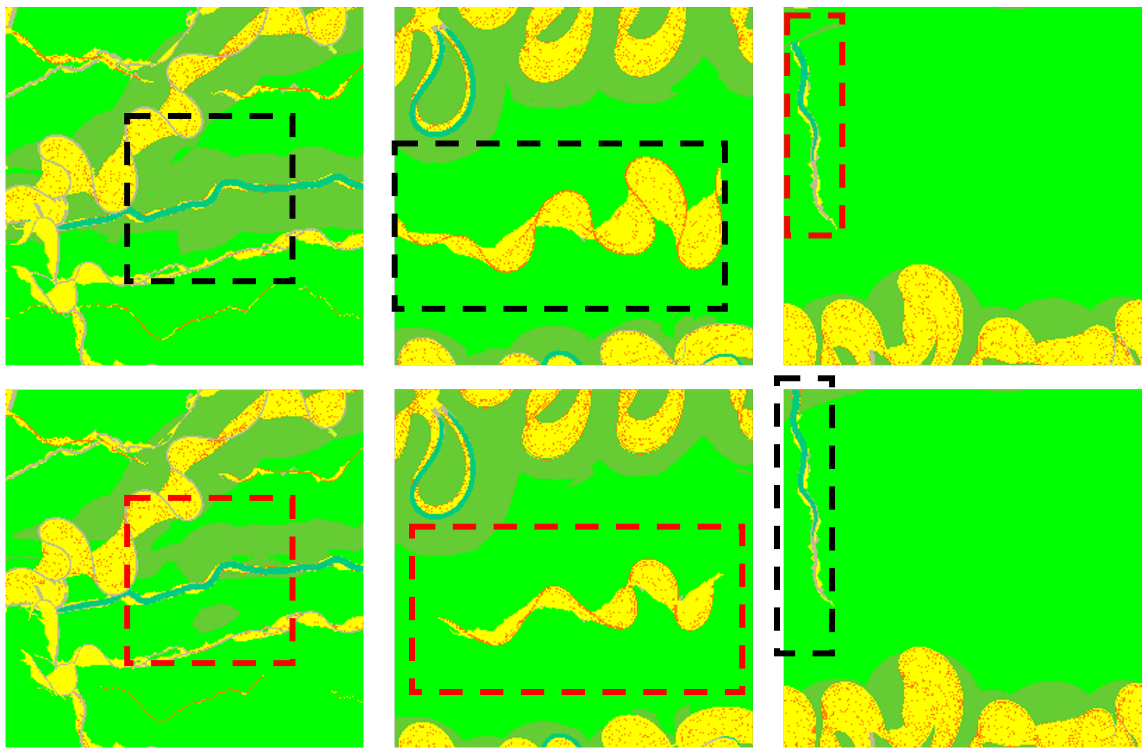


FIGURE 3.13: FLUMY 2D slices containing detached geo-body. The red dash line rectangular is the location of the detached geo-body. The black dash line rectangular shows the corresponding geo-body above or below the realisations with detached geo-body (red dash line rectangular).

requires the modeller/geologist to manually count the number of facies realisations containing a specific geologically unrealistic feature and then calculate the unrealistic realisations ratio to total realisations. The calculation of this occurrence-based score is given by Equation 3.5

$$Score = \frac{N_{feature}}{N_{total}} \tag{3.5}$$

where $N_{feature}$ refers to the number of facies realisations having a certain feature, and N_{total} is the total number of facies realisations used in this calculation. This occurrence-based score relies on many realisations to approach the probability of creating a certain feature and is subject to human interpretation. Still, it delivers high-level information about the model quality. Later chapters also use this occurrence-based score to evaluate GAN performance by calculating recurrent unrealistic features. This is because detecting the occurrence of a recurrent unrealistic feature is much more straightforward than visually judging if a change of GAN settings improves its generation quality. So, the lower the proposed score, the better the GAN performance in reducing a certain unrealistic feature.

3.3 Summary of Chapter 3

This chapter presented the process of creating the pre-canned training datasets, GAN River-I, used for GAN studies in the following chapters, starting from introducing the FLUMY software to selecting parameters and then to data pre-processing. For the convenience of evaluating GANs realisations using visualisation and connectivity, I decided to focus on low net-to-gross meandering fluvial models in this thesis. Considering the number of facies used in previous papers, I chose the three-facies version of GAN River-I in the comparison study to select a GAN variant as the baseline for the multi-facies modelling study. To avoid excess concentration on facies with very small proportions, I decided to use the seven-facies version of GAN River-I as the benchmark for the rest studies of GAN for multi-facies modelling.

Using this seven-facies GAN River-I dataset as the reference, this chapter carries quantitative and qualitative analyses, introducing methods used to evaluate GAN performance in later chapters. I adopted classical indicators used in geomodels description, including facies proportion and connectivity, as the quantitative measures. Then, I visualised the

GAN River-I dataset in a 2D space using a dimension reduction tool, UMAP, to show its model diversity. Also, this UMAP visualisation assists in assessing the model diversity in GAN-simulated ensembles in the following chapters. As GAN aims to deceive humans' interpretations, visual assessment is the most essential criterion in the GAN evaluation. I highlighted some odd patterns in GAN River-I raised by FLUMY's discretisation and came up with an occurrence-based score to assist in the qualitative analysis of GAN-simulated ensembles in later chapters.

Chapter 4

Fluvial GAN for Unconditional 2D Facies Modelling

This chapter studies generative adversarial networks (GANs)' learning capacity of 2D multi-facies meandering fluvial patterns from process-based models. This thesis starts from 2D instead of 3D modelling to avoid data scarcity and trapping in GANs' learning difficulties, and then extends GANs in 2D to tackle 3D challenges in Chapter 5. This chapter selects a baseline GAN by comparing different GAN variants succeeding in recent publications Chan and Elsheikh [2019], Laloy et al. [2018]. Then further research develops this baseline GAN to tackle identified difficulties to enhance GAN's performance in learning key geological features of meandering systems, named Fluvial GAN.

4.1 Comparison Study of Popular GAN Variants

GANs have various setups, favouring learning different geological models while developing a general solution to improve the learning capability for all GAN variants is very time-consuming and difficult. Because a general solution needs to be proven to work well with different datasets and GANs, which takes a long time to train, verify and test the proposed solution, so, this section doesn't aim to develop a general solution but uses the 3-Facies version of GAN River-I to compare different GANs for learning meandering fluvial patterns, which is used as the baseline model for further development. Different

GANs differ in model structures, loss calculations and training strategies. So, selecting a high-potential GAN variant as the baseline for further study from different GAN variants available needs a proper comparative study, which set-up is better for modelling meandering fluvial patterns. As earlier successful GAN applications to multi-facies modelling mostly use three facies models as their training dataset [Song et al., 2021b, Zhang et al., 2019b], this comparison study also uses the three facies models as the training dataset to avoid incompatible comparison.

4.1.1 Comparative Study Design

Four popular GANs are the candidates in this comparison study according to earlier successes in facies modelling and the computer vision domain [Chan and Elsheikh, 2019, Isola et al., 2017, Li and Wand, 2016, Zhang et al., 2019b]. The four candidates are DCGAN, WGAN, WGAN-GP and PatchGAN [Arjovsky et al., 2017, Gulrajani et al., 2017, Isola et al., 2017, Li and Wand, 2016, Radford et al., 2015]; see the review provided in section 2.3.3. PatchGAN is a successful application in image-to-image synthesis [Isola et al., 2017, Li and Wand, 2016]. This study introduces PatchGAN into unconditional data synthesis because of its discriminator's bright feature of tackling high-resolution images (see Figure 4.1). PatchGAN discriminator evaluates data in the patch level, a smaller region in the inputs as it doesn't have dense layers or convolutional networks in its deeper architecture [Li and Wand, 2016]. This feature reduces the number of learnable parameters, benefiting the training process [Isola et al., 2017, Li and Wand, 2016].

All GAN variants use the same generator with deep convolutional architecture and the default values of hyper-parameters in their original papers, except for PatchGAN [Arjovsky et al., 2017, Gulrajani et al., 2017, Li and Wand, 2016, Radford et al., 2015]. This study adopts the PatchGAN settings in Park et al. [2019] and adds an extra term, zero-centred gradient penalty, in its loss function to avoid mode collapse. Compared the one-centred gradient penalty in Gulrajani et al. [2017], the zero-centred and has a different centre value in its calculation, given by Equation 4.1

$$GP = \lambda \mathbb{E}_{\tilde{x}}[(\|(\nabla D)_{\tilde{x}}\| - c)^2] \tag{4.1}$$

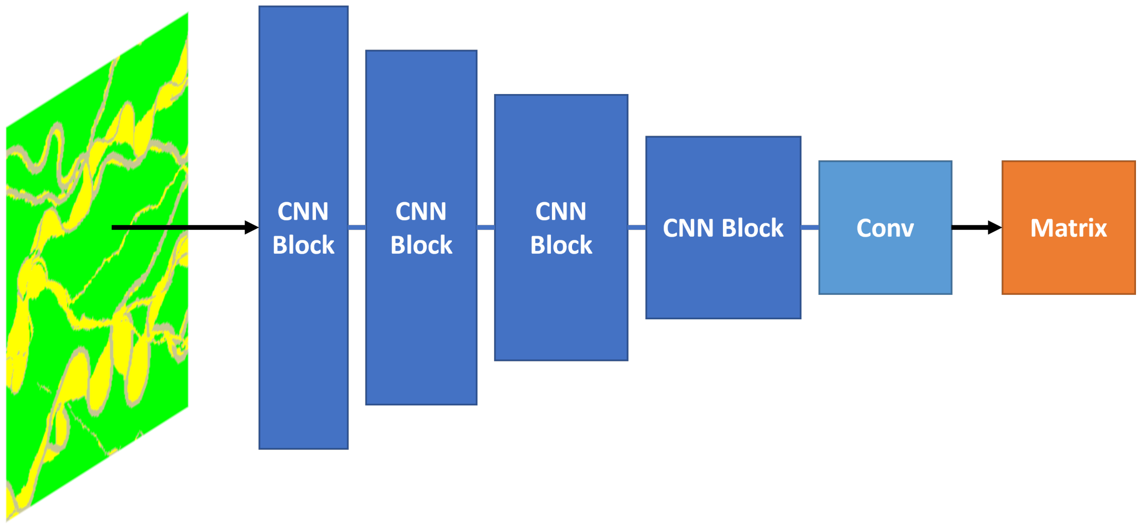


FIGURE 4.1: A schematic diagram of PatchGAN discriminator architecture. *CNNBlock* denotes a convolutional neural network block composed of a convolutional neural network, a normalization, and an activation. *Conv* is a convolutional layer. *Matrix* is the output identity in the format of matrix

where λ is a constant weight of gradient penalty, $\mathbb{E}[\cdot]$ is the mathematical expectation, ∇D is the gradient of the discriminator parameters, $\tilde{x}$ is a linear interpolation between real and fake data, and c is the centre value that equals to zero and one in the zero-centred gradient penalty [Thanh-Tung et al., 2019] and the one-centred gradient penalty [Gulrajani et al., 2017], respectively. So, the improved hinge loss function is given by Equation 4.2

$$Loss = \begin{cases} Loss_D + GP_{0-centered} \\ Loss_G \end{cases} \tag{4.2}$$

where $GP_{0-centered}$ is the zero-centred gradient penalty, $Loss_D$ and $Loss_G$ are hinge losses for the discriminator and the generator, respectively. The calculations of $Loss_D$ and $Loss_G$ are given by Equation 4.3 and 4.4, following the implementation in Lim and Ye [2017],

$$Loss_D = \mathbb{E}_{x \sim P_x}[\max(0, 1 - D(x)] + \mathbb{E}_{z \sim P_z}[\max(0, 1 + D(G(z)))] \tag{4.3}$$

$$Loss_G = -\mathbb{E}_{z \sim P_z}[D(G(z))] \tag{4.4}$$

where $G(\cdot)$ and $D(\cdot)$ are the sequential calculations of the generator and the discriminator in neural networks, and $\mathbb{E}[\cdot]$ is the mathematical expectation.

Table 4.1 summarises GANs settings used in this study. The original setting of every GAN candidate might not be optimal for generating meandering fluvial patterns, and further improvements on those GAN variants may achieve better results. However, considering the time cost, this study can only select one best-performing GAN and further develop it instead of making improvements for every GAN variant.

TABLE 4.1: Summary of GANs set up in the comparison study.

GANs Name	DCGAN	WGAN	WGAN-GP	PatchGAN
D architecture	DC	DC	DC	Patch
Normalization	Batch	Batch	Layer	Spectral instance
Loss Function	Cross entropy	Wasserstein	Wasserstein	Hinge loss
Extra term	None	Weight clipping	1-centred GP	0-centred GP
Optimiser	ADAM	RMSprop	ADAM	ADAM
Train D/G	1	5	5	1

DC denotes the deep convolutional discriminator. Patch denotes the PatchGAN discriminator. D denotes the discriminator. G denotes the generator. 1-centred GP denotes the one-centred gradient penalty. 0-centred GP denotes the zero-centred gradient penalty.

Apart from the differences in model architecture and optimisation, the data pre-processing, training dataset and epochs are the same for all GAN variants. The data pre-processing linearly rescale the encoded facies integers to [-1, 1], following earlier GAN applications [Chan and Elsheikh, 2019, Song et al., 2021b, Zhang et al., 2019b]. To make it easier to determine and pick a type of meandering fluvial model, this study separates the 3-Facies version of GAN River-I into two training datasets; one only includes high avulsion rate models from Groups 1 and 2, and the other comprises low avulsion rate models from Groups 4 and 5 in Table 3.3. The training for every GAN variant runs 100 epochs on each training dataset individually, yielding eight trained GAN models.

This study relies on visual analysis of GANs' generations according to qualitative criteria regarding the meandering pattern and GAN learning. The major criteria are the geological consistency between GANs' generations and their training dataset, particularly the curvilinear shape of the channel and point bar placement. In Figure 4.2, the red circle highlights the meandering channel's geometry, and the blue circle shows a typical example of point bar placement along the channel. As mentioned in section 2.3.3, GAN often surfers the mode collapse issue. So, the occurrence of mode collapse is also one of the qualitative evaluation criteria in this study.

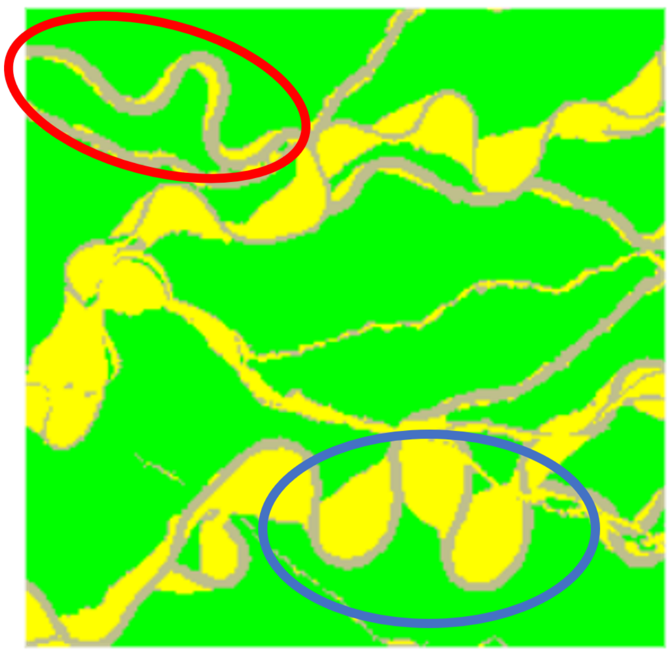


FIGURE 4.2: An example of the meandering pattern from GAN River-I showing the geological consistency criteria.

4.1.2 Result and Discussion

This study first evaluates GANs' performance in learning high avulsion rate models by randomly generating realisations from the four trained GAN variants. Visual comparison of the results based on the criteria mentioned in section 4.1.1 illustrates the learning performance of the four candidates (see Figure 4.3). DCGAN fails to capture the sinuous shape of the channel and the point bar placement. Also, DCGAN suffers a severe mode collapse problem that produces nearly the same realisations no matter how the input vector of the generator changes (see the DCGAN column of Figure 4.3). WGAN and WGAN-GP learn the elongated channel shape and place the point bar at the inner banks. However, their patterns of the channel across the domain are less likely plausible and sometimes they only create point bars on one side of the channel's inner bank instead of two sides (see the blue circle in Figure 4.3). PatchGAN achieves a more stable performance regarding the channel pattern and point bar placements, resembling fluvial channels meandering upstream to downstream.

Then, the second evaluation compares the four GANs' capability of reproducing low avulsion rate models in the same way (see Figure 4.4). DCGAN only learns the local spatial

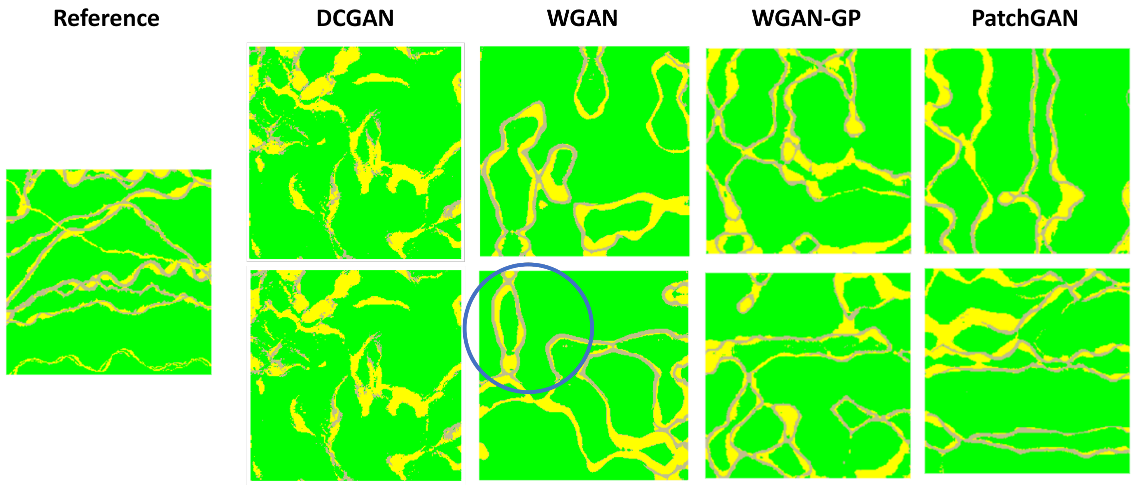


FIGURE 4.3: Random generations from different GANs trained with high avulsion rate meandering models.

correlations between the three facies: channel, point bar and background shale. DCGAN places the channel along the point bars' margin while doesn't reproduce the channel shape. WGAN reproduces the channel shape but fails to capture the spatial relations fully. It still tends to create point bars along one side of the channel's inner bank, which is likely the inner side of an object (see the blue circle in Figure 4.4). WGAN-GP performs similarly to WGAN, and PatchGAN performs the best in learning the low avulsion rate models as well.

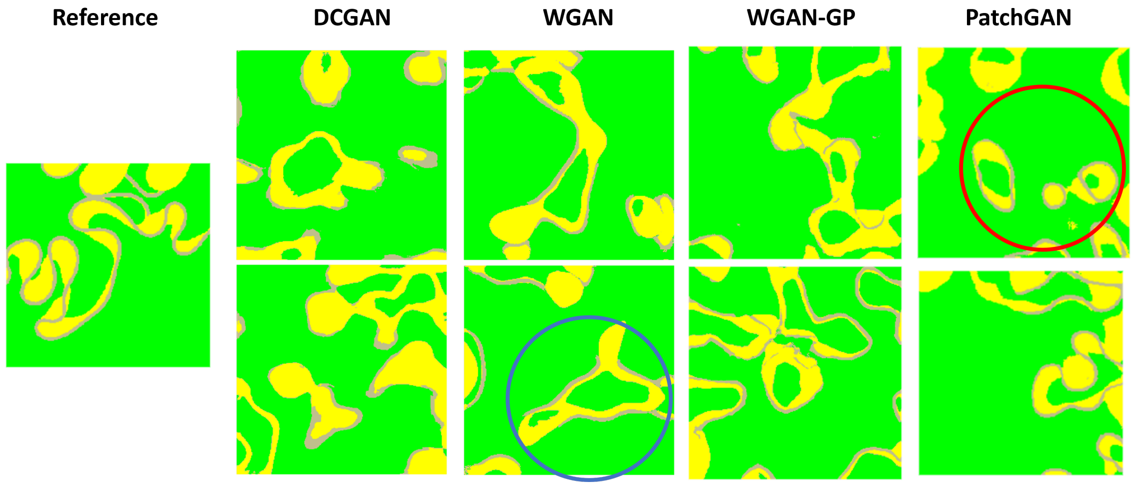


FIGURE 4.4: Random generations from different GANs trained with low avulsion rate meandering models.

All four GAN variants experience difficulty in reproducing the more complex geological patterns from the process-based model, FLUMY. Those GANs' realizations contain unrealistic features, even PatchGAN, summarised in Table 4.2. Although PatchGAN outperforms others in this comparison study, there are still some geologically unrealistic features remaining. One critical unrealistic feature of PatchGAN is the 'closed channel' pattern (see the red circle in Figure 4.4). FLUMY realisations contain some reasonable loop patterns (see Figure 4.11 in later Section 4.2.3) while the 'closed channel' highlighted here presents different features. The 'closed channel' here means a fluvial channel created by GAN, which displays isolated circles/loops instead of a reasonable meandering character. All four GAN variants suffer this problem to some degree in learning both high and low-avulsion rate models, creating 'closed channels' or similar isolated circles with an even worse representation of local features, which are likely a training difficulty caused by a local minimum trap. So, the 'closed channel' issue is problematic for all four GAN variants but is a crucial challenge for PatchGAN because its discriminator provides feedback based on the local information at the patches scale. Section 4.2.3 will discuss reducing this unrealistic feature.

TABLE 4.2: Summary of the qualitative comparison for reproducing key geological features and GANs stability.

GANs Name	DCGAN	WGAN	WGAN-GP	PatchGAN
Channel sinuous shape	×	✓	✓	✓
'Closed channel' pattern	✓	✓	✓	✓
Point bar aside channel	✓	✓	✓	✓
Point bar on both sides of inner bank	×	×	×	✓
Mode collapse	✓	×	×	×

✓ denotes Yes. × denotes No.

Data size and complexity lead to GANs' failure to capture the complex geometries of the meandering fluvial channels and associated facies. This study trained GANs on higher resolution 2D data containing 256×256 pixels compared to earlier successful applications using 64×64 or 128×128 pixels images [Chan and Elsheikh, 2019, Song et al., 2021b], making it hard for GANs to create bigger size images matching the patterns of the training data without mode collapse. PatchGAN has a much lighter discriminator and focuses on patterns in a smaller region instead of the whole image, reducing the learning stress of the generator by narrowing down the size of patterns to be matched.

On the other hand, the meandering fluvial models in GAN River-I have meandering channels with various shapes and asymmetric sinuosity, and non-constant point bar geometries. Channels in earlier publications mostly have constant sinuosity and shape [Chan and Elsheikh, 2019, Song et al., 2021b]. Though some publications set different channel sinuosity to create training datasets, the curvatures within a training image remain unchanged, and the facies linearly transit from channel to levee, then to background [Song et al., 2021b, Zhang et al., 2019b]. Channels in this study, however, have different curvatures within the same channel, trigger meander cut-offs when maturely developed, and cross over each other to form more complex patterns. The increased data complexity gives GANs a sterner test, resulting in all four GAN variants struggling to learn plausible geological patterns at different levels of realism reproduction.

4.2 Fluvial GAN

This study uses the 7-Facies version of GAN River-I to test GANs' performance in creating multi-facies meandering fluvial models. To the best of my knowledge, this training dataset contains more facies than earlier publications, increasing the complexity by adding more details to geobodies. In addition to the 'closed channel' issue mentioned in the last section, applying GAN to multi-facies modelling introduces another issue, named 'mislabelling'. 'Mislabelling' appears when GAN uses wrong encoded facies to create geobodies, drawing a value belonging to another facies but numerically close to the encoded values of the correct facies, for example, creating a muddy abandoned channel assigned with facies whose encoded values are between mud plug and point bar, e.g. overbank, crevasse splay, etc (see the upper red circles in Figure 4.6 e).

This section introduces Fluvial GAN, a further developed GAN model, to reproduce multi-facies meandering fluvial patterns created in various depositional settings. Using Patch-GAN, the winner of our comparison study, as the baseline, Fluvial GAN has three specific improvements: (1) an embedded One-Hot Encoder to treat facies as nominal data, (2) a proposed Hybrid-discriminator to better capture fluvial patterns, and (3) a variant hinge loss to avoid mode collapse. Both qualitative and quantitative methods compare Fluvial GAN and two more standard cases in reproducing the geological features represented by GAN River-I. Fluvial GAN largely reduces the occurrence of the 'mislabelling' and 'closed channel' issues. This study employs UMAP (see section 3.2.2.2) to visualise the diversity of generated realisations by comparing the ensemble with the training dataset in the UMAP 2D space. Fluvial GAN well covers the uncertainty displayed by UMAP projected GAN River-I.

4.2.1 Problems of Applying Baseline to Multi-facies Modelling

As mentioned in Section 4.1.1, the PatchGAN with an improved loss function outperforms others and therefore, becomes the baseline to multi-facies modelling. Data pre-processing continues to use linear scaling, and PatchGAN settings remain the same as Table 4.1, except for the optimiser (see Figure 4.5). This study selects AdamW [Loshchilov and Hutter, 2017] instead of ADAM as the optimiser because the standard Fluvial GAN model in later sections uses AdamW based on the comparative trial runs. Comparing the optimisers' impact on learning fluvial patterns is beyond the scope of this thesis as the exploration is not exhaustive, and its impact is less significant than the improvements on the model architecture. Even though the settings of the optimiser didn't dominate GAN's performance in this case, this study still needs to keep the used optimiser consistent for the ablation studies of further improvements. An ablation study infers a user-selected element's contribution to an artificial intelligence system by comparing the results with and without this component while keeping other settings the same.

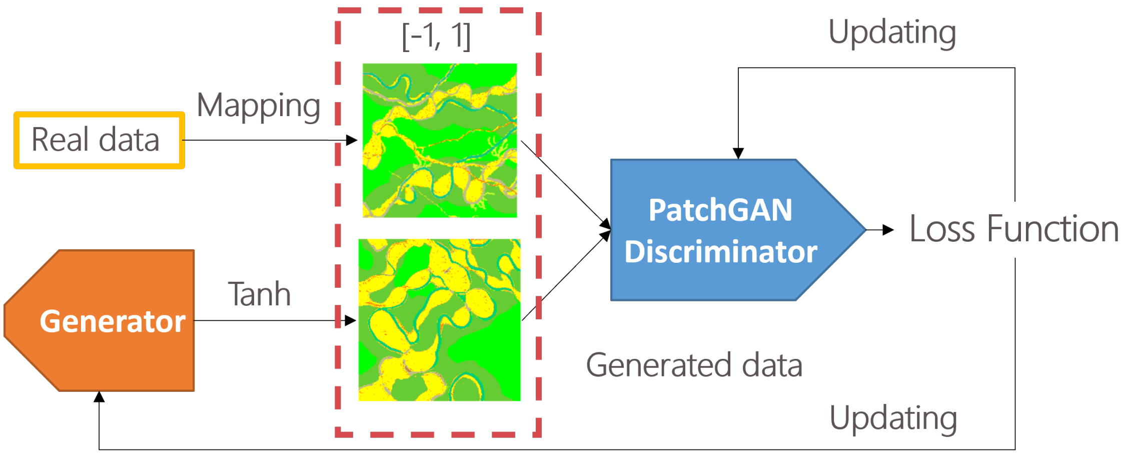


FIGURE 4.5: A schematic diagram of PatchGAN workflow for multi-facies modelling.

The results of the baseline present clear challenges of applying PatchGAN to simulate fluvial geology (see Figure 4.6). Though most realisations show meandering shapes of the channel and associated facies, the reproduced meandering fluvial patterns have several issues. The first problem is the facies models from the baseline are visually incomparable to the image quality of its training dataset, showing indistinct shapes of facies. For instance, the sand plug breaks up along the channel in GAN realisations (see Figure 4.6 b), while it is unbroken in GAN River-I. The channel fills often do not present the proper meandering shape (see Figure 4.6 e). Facies sometimes also follow implausible ordering; for example, 'overbank' often surrounds the mud plug facies when it should border on the point bar or levee (see Figure 4.6 a). Those inaccurate patterns are the so-called 'mislabelling' in this study. Some GAN generations show unrealistic features, e.g. placing levee facies on one side of meander-belts (see Figure 4.6 d) or the channel or meander-belt forming an isolated circle (see Figure 4.6 f). The latter pattern is the 'closed channel' issue mentioned earlier, which is easy to spot, but apparently implausible.

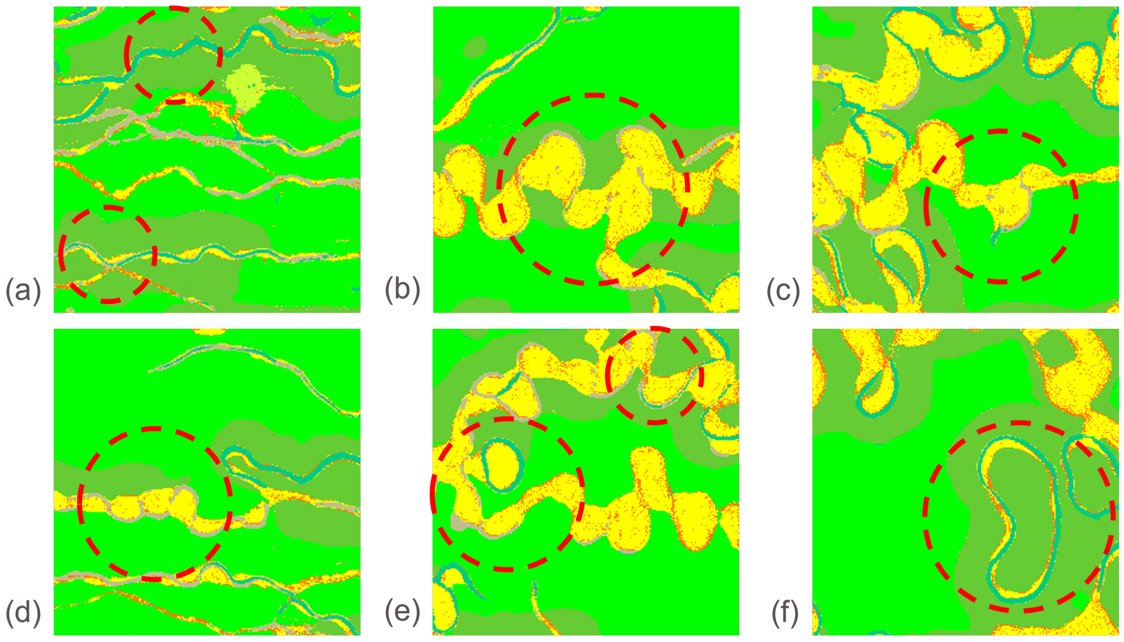


FIGURE 4.6: 2D realisations from Baseline GAN. (a)(b)(c) Examples of GAN mimicking high to low avulsion rate fluvial models. (d) An example of GAN creating discontinuous channel shape and inappropriate facies transition. (e) An example of GAN realisation with an apparent 'mislabelling' problem. (f) An example of GAN generating inaccurate 'closed channel' patterns.

4.2.2 Appropriate Data Pre-processing for Categorical Data

One improvement on the baseline embeds the One-Hot Encoder, a data pre-processing algorithm that transforms data from category to numeric, into the GAN framework to handle the ‘mislabelling’ issue. One-Hot Encoder encodes the 2D training facies data to a 3D matrix that adds an extra dimension to indicate which facies exist at locations (see section 2.3.1.1 for the details of One-Hot Encoder). To integrate the One-Hot Encoder, SoftMax replaces Tanh as the last activation function of the generator whose output spectral dimension is seven, equalling the total number of facies. SoftMax indicates the relative probability of categories at locations (see section 2.3.1.2 for the details of SoftMax). This change makes the generator’s output consistent with the result processed by the One-Hot Encoder, avoiding the discriminator processing inputs with different sizes. The rest remains the same as Baseline, including the training epoch.

GAN with the One-Hot Encoder produces more distinct individual geometries that are visually comparable to FLUMY realisations (see Figure 4.7). The One-Hot Encoder significantly improves GAN’s generation quality based on a visual comparison between the Baselines with and without the embedded One-Hot Encoder. Sand and mud plug facies filled in a channel now appear as unbroken meandering shapes (see Figure 4.7 a, 4.7 b, 4.7 c). Facies transitions become more accurate along the channel centreline shown in GAN realisations (see Figure 4.7 a, 4.7 b, 4.7 c, 4.7 d). However, the ‘closed channel’ pattern still exists in many GAN realisations (see Figure 4.7 e, 4.7 f), meaning this is still an unsolved problem through the assistance of the One-Hot Encoder.

A comparison between the Baselines with and without the embedded One-Hot Encoder implies that the ‘mislabelling’ issue may result from inappropriate data pre-processing. GAN realisations get visually more akin to FLUMY realisations after changing the data pre-processing from linear mapping to the One-Hot Encoder; compare the results shown in Figure 4.6 and Figure 4.7. Many Baseline GAN realisations have the ‘mislabelling’ issue, displaying some facies in the other facies’ place, whose encoded values produced by data pre-processing are close. Baseline GAN linearly scales a large number of facies (categories), leaving a small interval between encoded facies in continuous values, raising the ‘mislabelling’ problem. For instance, Baseline GAN inserts crevasse splay and overbank

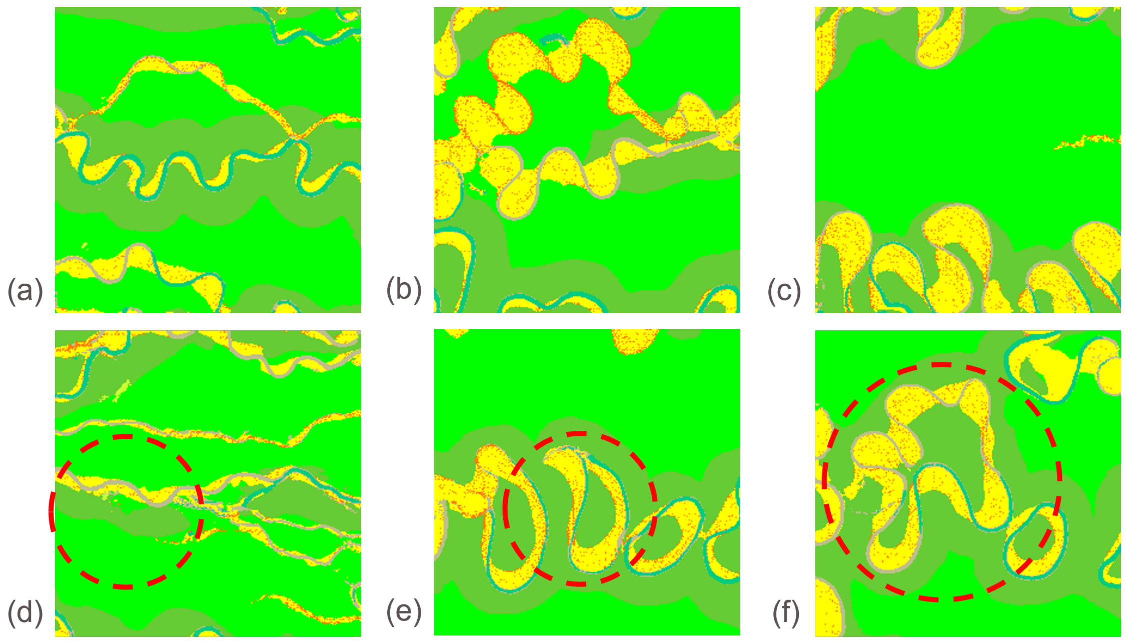


FIGURE 4.7: 2D realisations from Baseline GAN with One-Hot Encoder. (a)(b)(c) Good examples of GAN mimicking high to low avulsion rate fluvial models. (d) An example of GAN creating inaccurate facies transition. (e)(f) Examples of GAN generating inaccurate 'closed channel' patterns.

facies between the mud plug and other facies, e.g. point bar and levee (See Figure 4.6 a), and creates the sand plug facies in the point bars' place, resulting in a broken channel (see Figure 4.6 d) or wrong channel geometries (see Figure 4.6 e).

The One-Hot Encoder is a more appropriate choice to process facies without rigorous ordering for GAN learning. Though facies may obey an order following certain rules in the process of integer encoding, e.g. ordinal encoding, one order is often inadequate to describe the relationships between facies in all rock properties as they are non-linear. For instance, sand plug facies has a larger grain size and higher permeability than mud plug facies. In contrast, the channel lag has a larger grain size than sand plug facies while sometimes is less permeable because of poor sorting or impermeable if well cemented by carbonate [McKie and Audretsch, 2005]. Thus, unless there is a firm ordering, the modeller should treat facies as order-independent categories (nominal data), which is the idea of the One-Hot Encoder and what geostatistics has been doing for decades. Hence, the One-Hot Encoder naturally suits better to handle multi-facies modelling, consistent with the fact that GAN with the One-Hot Encoder has fewer 'mislabelling' issues than the Baseline.

4.2.3 Fluvial GAN Learns Meandering Patterns Better

This study designs a GAN variant with better stability for meandering fluvial facies modelling, named the Fluvial GAN (see Figure 4.8). The new design addresses the identified problems in previous sections: the 'mislabelling' and 'closed channel' issues. Fluvial GAN uses the conventional GAN learning framework but has three unique improvements: (i) a One-Hot Encoder to prevent the generator from drawing naturally ordered values to facies without a natural order (please refer to section 4.2.2), (ii) a modified discriminator structure named the Hybrid-discriminator, reducing 'closed channel' patterns' occurrence and (iii) an improved hinge loss with a zero-centred gradient penalty term to prevent mode collapse, ensuring Fluvial GAN's generations containing diverse geological patterns from the GAN River-I (please refer to section 4.1.1).

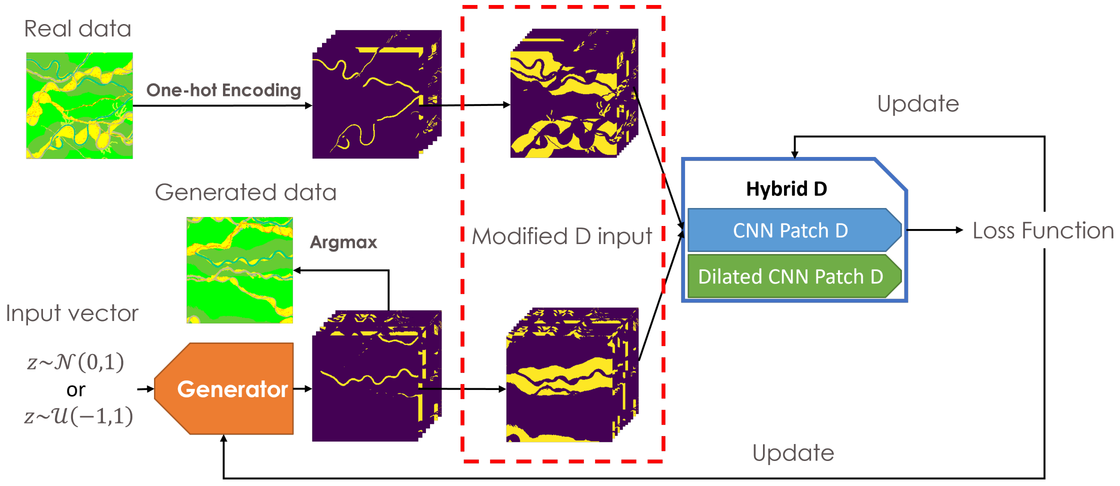


FIGURE 4.8: A schematic diagram of Fluvial GAN training workflow. One-Hot Encoding refers to the pre-processing of raw facies data using the One-Hot Encoder, which produces a set of maps indicating the presence of facies at locations. The generator outputs are a set of probability maps of facies. Argmax converts the probability maps to a facies realisation by assigning the facies with the highest probability at locations. Hybrid D denotes the Hybrid-discriminator composed of two convolutional neural networks-based classifiers. Modified D input within the red dash-lined rectangle denotes an optional operation to modify data before feeding it to the discriminator, which changes the number of probability maps used as the input of the discriminator.

The Hybrid-discriminator in Fluvial GAN is another improvement on the baseline GAN, which modifies the architecture of the PatchGAN discriminator to prevent the generator from creating geologically unrealistic features. Though having a similar idea of increasing the discriminator's receptive field as the multi-scale discriminator [Park et al., 2019, Wang et al., 2018b], the Hybrid-discriminator captures different receptive fields by varying the convolutional layers' dilation rates instead of directly resizing the input. The hyper-parameters of the Hybrid-discriminator, e.g. the number of discriminators and the dilation rate in each layer, presented in this study is a trial-and-error result, leaving the architecture optimisation as future research. The Hybrid-discriminator in Fluvial GAN consists of two PatchGAN discriminators with different dilation rate configurations that capture the geological patterns *of different sizes*. One discriminator only contains standard convolutional layers, while the other has several dilated convolutional layers. Please refer to section 2.3.1.2 for the overview of convolution and dilated convolution. Such a Hybrid-discriminator structure is essential to introduce better reproduction of multi-scale meandering patterns.

Therefore, Fluvial GAN's model structure comprises a modified DCGAN generator and two varied PatchGAN discriminators. Table 4.3 presents the details of the Fluvial GAN generator. The two PatchGAN-based discriminators' hyper-parameters are summarised in Table 4.4 and 4.5, respectively. Following Park et al. [2019], both the generator and the discriminator in Fluvial GAN use spectral normalisation to stabilise GAN training, which is a weight normalisation algorithm applied to GANs' learnable parameters (weight values) [Miyato et al., 2018].

Table 4.3: Fluvial GAN Generator Architecture

Layer	State size
Linear (128, 32768)	(Batch size, 32768)
Reshape	(Batch size, 512, 8, 8)
NN Block, Up sample (2)	(Batch size, 512, 16, 16)
NN Block, Up sample (2)	(Batch size, 256, 32, 32)
NN Block, Up sample (2)	(Batch size, 128, 64, 64)
NN Block, Up sample (2)	(Batch size, 64, 128, 128)
NN Block, Up sample (2)	(Batch size, 32, 256, 256)
NN Block, 3X3Conv-7, SoftMax	(Batch size, 7, 256, 256)

3X3Conv-7 denotes 3X3 convolutional layer with seven filters. NN Block refers to the neural networks sequentially composed of a batch normalisation, a leaky ReLU, and a 3X3 convolutional layer.

Fluvial GAN has an optional process that modifies the discriminator's inputs in spectral dimension (the data science community often calls it 'channel dimension', but it is confusing in this study) to introduce extra geological information of the sedimentary relationships between facies into the discriminator (see the Modified D input in Figure 4.8). This process

TABLE 4.4: CNN-based PatchGAN Discriminator in Fluvial GAN Hybrid Discriminator

Layer	Conv Parameters (kernel size, stride, dilation)
Conv-32, Leaky ReLU	(4, 2, 1)
Conv-64, IN, Leaky ReLU	(4, 2, 1)
Conv-128, IN, Leaky ReLU	(4, 2, 1)
Conv-256, IN, Leaky ReLU	(4, 1, 1)
Conv-1	(4, 1, 1)

IN denotes instance normalisation.

TABLE 4.5: Dilated CNN-based PatchGAN Discriminator in Fluvial GAN Hybrid Discriminator

Layer	Conv Parameters (kernel size, stride, dilation)
Conv-32, Leaky ReLU	(4, 2, 1)
Conv-64, IN, Leaky ReLU	(4, 2, 2)
Conv-128, IN, Leaky ReLU	(4, 2, 2)
Conv-256, IN, Leaky ReLU	(4, 1, 2)
Conv-1	(4, 1, 1)

IN denotes instance normalisation.

excludes the background-indicator layer, overbank facies, from the discriminator's input because background facies is unnecessary to present a particular shape and is a result of other facies' geometries instead. Then the rest six facies-indicator layers concatenate four grouped facies layers (see Table 4.6), composing the modified inputs of the discriminator. Each grouped facies preserves the overall shape of a geo-body that contains several original facies with different petrophysical properties. Each original facies can exist in more than one grouped facies. Thus, the discriminator's input has ten layers in the spectral dimension equalling the total number of facies-indicators, including the six original facies-indicator layers and the four grouped facies-indicator layers.

TABLE 4.6: Summary of the four grouped facies (geobodies) used as extra indicator layers

Grouped facies (Geobodies)	Original facies from FLUMY
Channel deposits	Sand plug, mud plug
Lateral accretion deposits	Channel lag, point bar
Meander-belt deposits	Channel lag, point bar, sand plug, mud plug
Levee deposits	Crevasse splay, levee

The training process of Fluvial GAN runs 200 epochs to learn the meandering fluvial facies distribution from GAN River-I. The Fluvial GAN's training process uses AdamW [Loshchilov and Hutter, 2017] as the optimiser whose learning rate is 0.0002, and betas values are 0.5 and 0.9. The training and testing of Fluvial GAN both use a single GPU, RTX3090, to benchmark the speed. A single RTX3090 needs about 23 hours to train Fluvial GAN for 200 epochs and can produce around 16000 facies models per minute using a pre-trained Fluvial GAN. See Figure 4.9 for the curves of loss values during training.

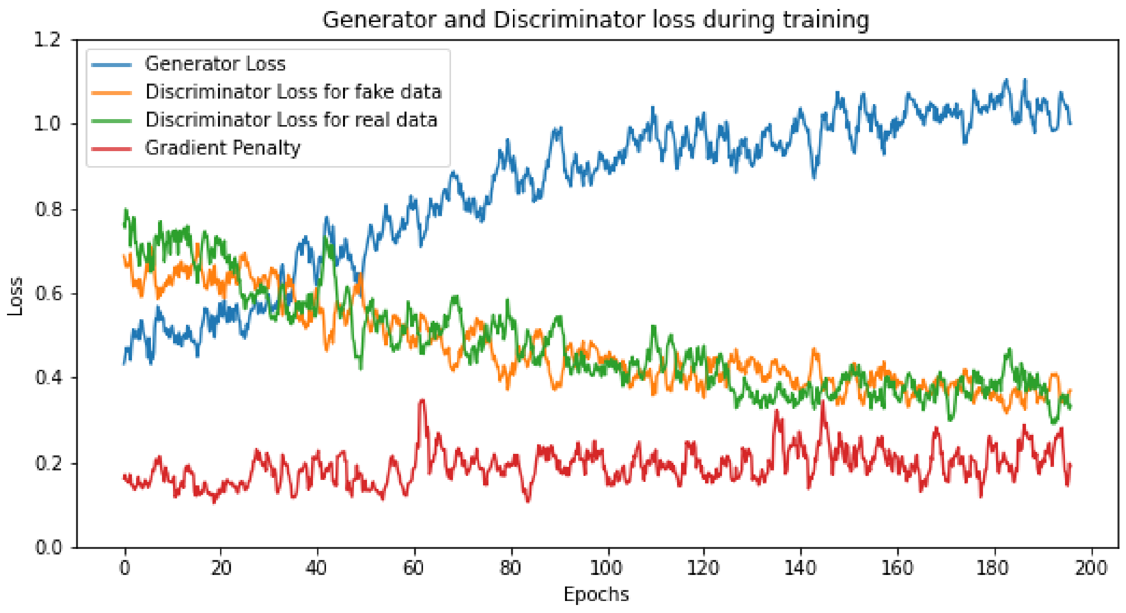


FIGURE 4.9: Loss curves of Fluvial GAN during training.

Fluvial GAN performs better based on visual comparison with the training dataset and by evaluation regarding geological realism (see Figure 4.10). Fluvial GAN reproduces the facies geometries and transitions observed in GAN River-I and occurs 'closed channel' patterns around *ten times fewer* than the baseline and the one-hot encoder embedded case. Like other GAN variants, the Fluvial GAN also creates artefacts and fails to learn certain geological patterns, e.g. the channel-filled facies transition along the abandoned channel, resulting from an intricate sediment transport mechanism in the abandoned channel affected by the active channel. Occasionally, facies models from Fluvial GAN suffer the 'gridding effect' issue due to dilated convolution (see section 2.3.1.2 for the details), displaying one or more facies with a gridding texture on another facies (see Figure 4.10 d). Some realisations from the Fluvial GAN present abandoned channels with frequent changes in the channel-filled facies (sand plugs and mud plugs) downstream (see Figure

4.10 e). This pattern does not exist in GAN River-I, where the sand plug facies transit to mud plug facies only once after channel abandonment if the active channel isn't across. In rare cases, Fluvial GAN generates detached small objects with plausible local features (see Figure 4.10 f), but when treated as part of the whole picture, they are artefacts of facies models as they lack reasonable evidence of their existence, unlike the detached objects in 2D data from GAN River-I that has slices below or above it (see Figure 3.13).

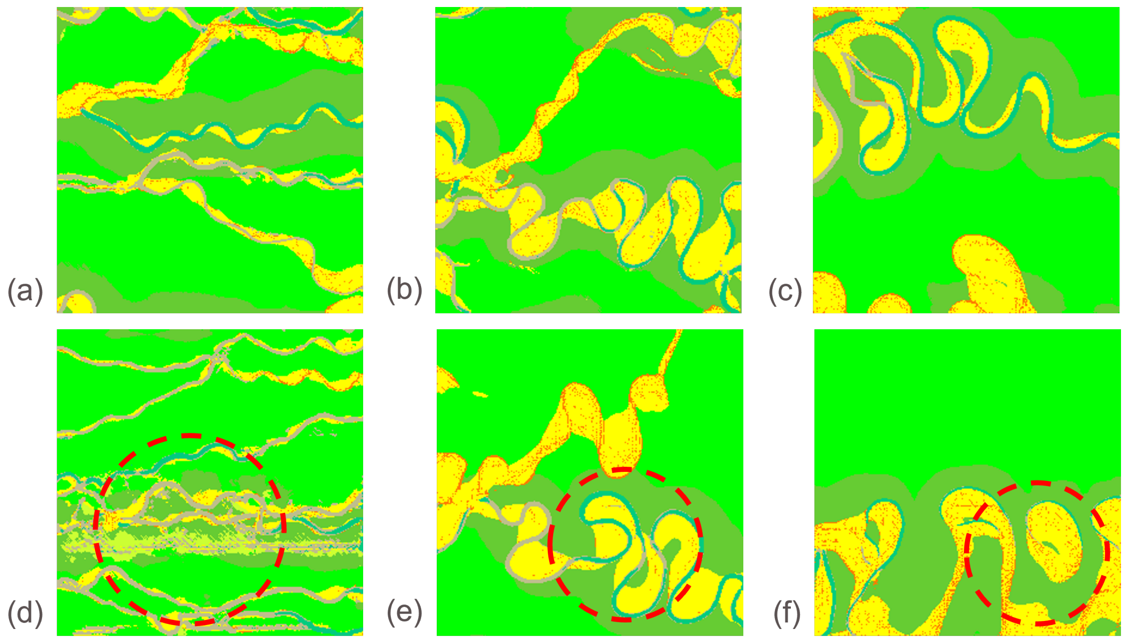


FIGURE 4.10: 2D realizations from Fluvial GAN. (a)(b)(c) Good examples of GAN mimicking high to low avulsion rate fluvial models. (d) An example of GAN suffering the 'gridding effect'. (e) An example of GAN changing channel fill facies frequently. (f) An example of GAN-generating artefacts.

The 'closed channel' issue is the toughest and the most conspicuous pattern in GANs generations, which is a geologically unrealistic feature. All three cases in this section produce this kind of 'closed channel' pattern: the channel or meander-belt connected end-to-end forms a detached circle (see Figures 4.6 f, 4.7 e, 4.7 f). The 'closed channel' here is an inaccurate reproduction of plausible meandering fluvial patterns observed in GAN River-I and may result in unreliable connectivity, which impacts the predictions of the flow response and sweep efficiency from any dynamic simulations based on those facies models.

FLUMY has several processes that result in some loop patterns, e.g. meander cut-offs and multiple channels deposited in the same elevation while crossing each other (see Figure

4.11). However, those ‘closed channel’ patterns produced by GANs differ from the plausible fluvial loops in GAN River-I. Those ‘closed channels’ disjoin the meander belt both upstream and downstream, lacking supporting clues about how they form.

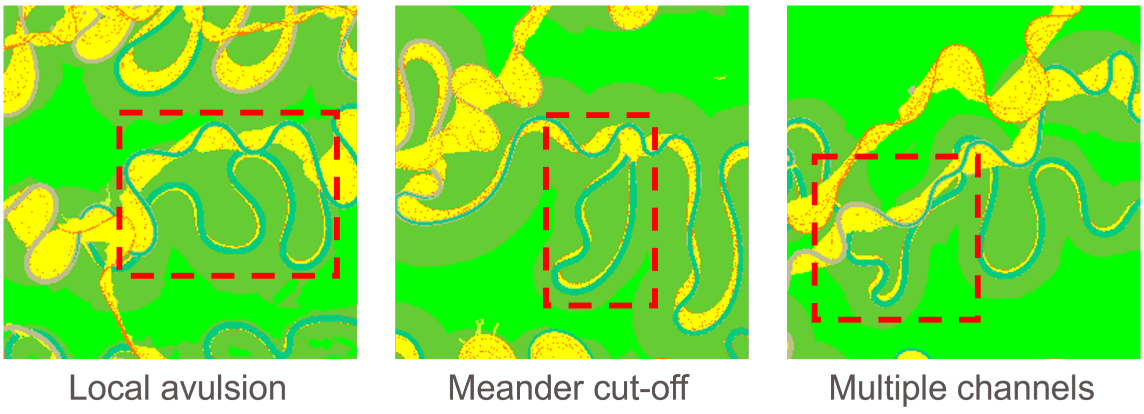


FIGURE 4.11: FLUMY simulations containing loop patterns.

The ‘closed channel’ issue may derive from the limitation of the convolutional layer’s local receptive field since the ‘closed channel’ pattern is plausible locally but implausible globally (see Figures 4.6 f, 4.7 e, 4.7 f). As mentioned in section 4.1.2, PatchGAN focuses on learning texture-level structure within small regions in training data, making how to avoid the ‘closed channel’ issue challenging. While other GAN variants in section 4.1.2 also suffer from this problem according to the comparison study (please refer to section 4.1.2 for the details). GANs learning is likely to stick in this sub-optimal state.

The idea behind the Hybrid-discriminator solution in Fluvial GAN is to broaden the discriminator’s receptive field while not dramatically increasing the number of learnable parameters. Classical methods of expanding the receptive field, e.g. deeper architecture or larger kernel size, lead to more learnable parameters and, consequently, raise the overfitting risk. In those cases, the discriminator quickly finds a way to determine where its inputs are from, real or fake data. In other words, the discriminator surpasses the generator and no longer provides a useful response to improve the generator’s performance. Therefore, Using dilation to expand the receptive field is a reasonable choice as the dilation doesn’t increase the number of learnable parameters (please refer to section 2.3.1.2 for more details). As presented in Wang et al. [2018a], using dilated convolution can lead to a serious ‘gridding effect’ problem (see Figure 2.19). Thus, following the hybrid dilation rates idea in Wang et al. [2018a], an extra GAN case uses different dilation rates in the discriminator of the baseline GAN with One-Hot Encoder to test the hybrid dilation rates discriminator's performance. However, the occurrence and level of the 'gridding effect' issue are still not at an acceptable level (see Figure 4.12).

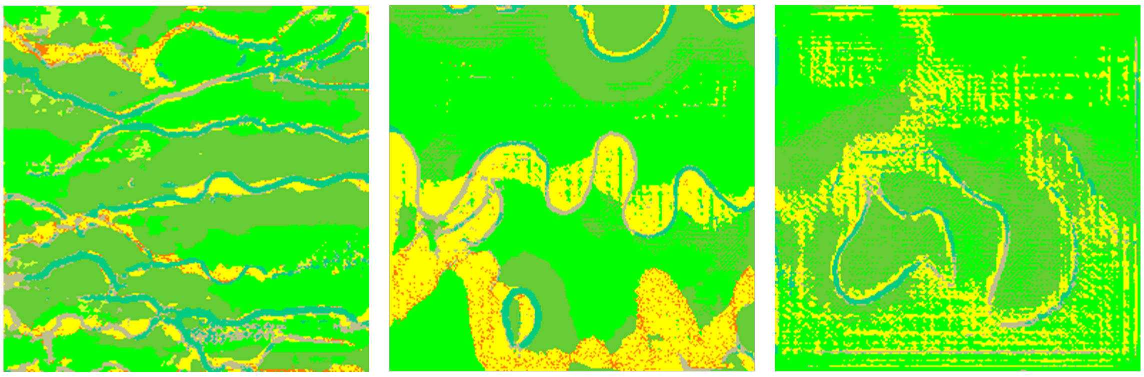


FIGURE 4.12: Examples of realisations suffering strong 'gridding effect' from a Fluvial GAN variant that only uses a dilated convolution-based discriminator with a hybrid dilation setting. This discriminator is the same one in Table 4.5.

The proposed Hybrid-discriminator decreases both the 'closed channel' and 'gridding effect' to a low occurrence but does not fully erase them. This study uses the proposed occurrence-based score (see Equation 3.5) to evaluate Fluvial GAN regarding the 'closed channel' problem based on 1,000 random generations, which is manually computed (see Table 4.7). The score of the 'closed channel' occurrence reduces from 10% to about 1% by using the Fluvial GAN design. The 'gridding effect' occurrence in Fluvial GAN realisations also decreases to about 2%, much lower than the occurrence score of the variant that only has a dilated convolution-based discriminator, which is about 30%.

TABLE 4.7: Occurrence-based indicator of 'closed channel'

Case	'Closed channel' Occurrence
Baseline	11.2%
Baseline with One-Hot Encoder	9.8%
Fluvial GAN	0.7%

4.2.4 Analysis of Fluvial GAN

This section evaluates Fluvial GAN's performance qualitatively and quantitatively by randomly generating 1,000 realisations. Like the quantitative analysis of GAN River-I, this study uses the sand proportion and connectivity measures to evaluate the Fluvial GAN simulated set and compares it with GAN River-I's true subset in section 3.2.2.

The first assessment plots the Fluvial GAN realisations and the 'reference' (1,600) true subset of GAN River-I based on their sand connectivity against sand proportion to illustrate Fluvial GAN's performance regarding these two measures. The red and blue dots, representing the data from Fluvial GAN and GAN River-I, broadly overlap in Figure 4.13, illustrating that the Fluvial GAN ensemble covers the ranges of sand proportion and connectivity represented by the reference FLUMY ensemble. Table 4.8 recaps the statistics of the two (Fluvial GAN set and the true subset) realisation ensembles' sand proportion and connectivity.

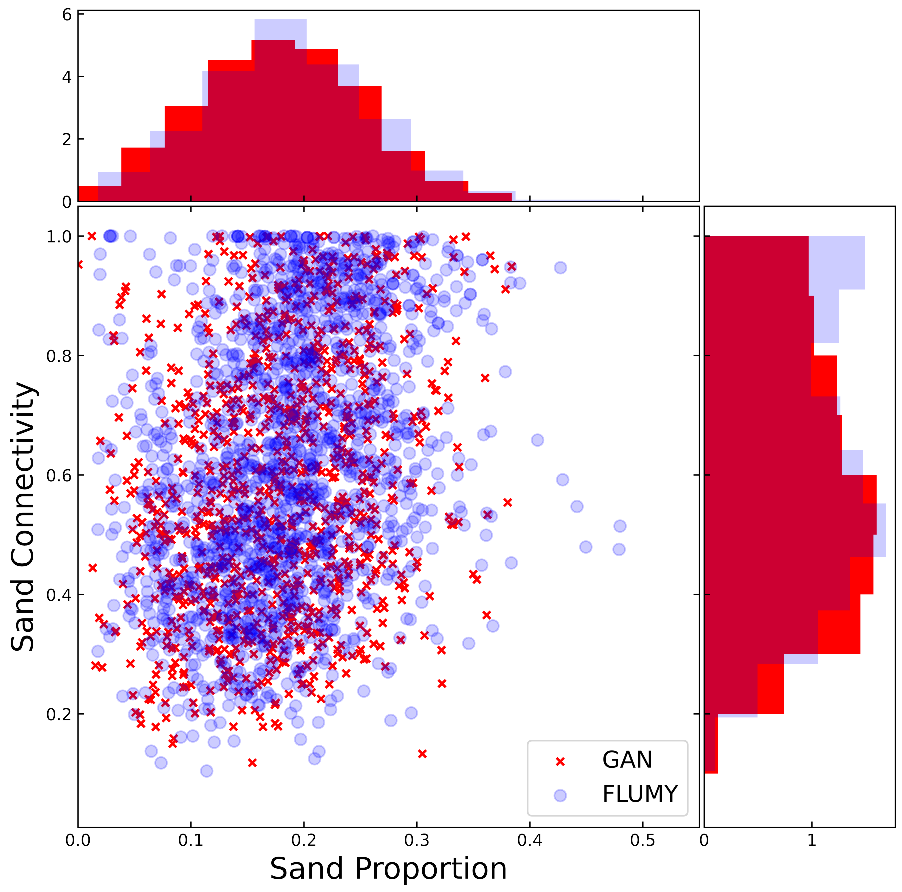


FIGURE 4.13: Plot of Fluvial GAN set's Sand Connectivity against Proportion.

TABLE 4.8: Statistics of Fluvial GAN set's Sand Connectivity and Proportion

Data Set	Sand Proportion		Sand Connectivity	
	Mean	Variance	Mean	Variance
FLUMY Subset	0.19	0.005	0.62	0.049
Fluvial GAN	0.18	0.005	0.58	0.047

Then an associated evaluation calculates all 2D realisations' sand connectivity probability from the Fluvial GAN set vs the FLUMY training data set. Figure 4.14 is a connectivity probability plot comparing all realisations from the Fluvial GAN and the FLUMY subset. Fluvial GAN shows a wide range of connectivity probability along lag distance (green curves) that covers the uncertainty of connectivity probability represented by the FLUMY set (red curves). To better compare the Fluvial GAN set and the FLUMY subset, Figure 4.14 uses lag box plots to indicate the 75%, medium and 25% cases of a realisation ensemble. Both the mean values (solid line) and the box plots (dash-line) of the two probability curve sets indicate the Fluvial GAN results well match the FLUMY subset curves (see the plot on the right side of Figure 4.14).

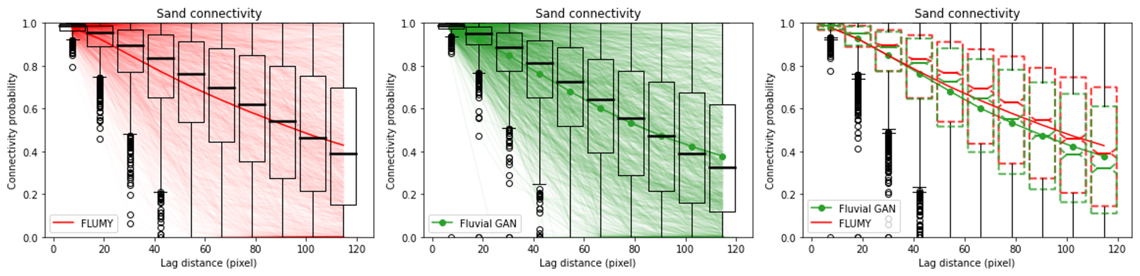


FIGURE 4.14: Sand connectivity probability curves. Green curves are realisations from Fluvial GAN. Red curves are realisations from the FLUMY training dataset. The solid lines are mean values, and the black box and dash-lined boxes indicate 75%, medium, and 25% of the data sets.

The second assessment evaluates the model diversity of the Fluvial GAN-simulated ensemble to demonstrate that the Fluvial GAN successfully avoid mode collapse, which means it can reproduce diverse types of geological patterns seen in GAN River-I. This assessment uses the same UMAP mapper used to analyse GAN River-I (see section 3.2.2.2 for the details) to visualise the ensemble realisations from Fluvial GAN in the same 2D metric space [McInnes et al., 2018]. Each dot in the UMAP metric space represents a 2D facies realisation (See Figure 4.15), providing a straightforward visual comparison of the model diversity between the Fluvial GAN realisations set and the subset of GAN River-I.

For the convenience of checking which type of realisation resulted from different avulsion rates exit, the whole Fluvial GAN sets (red dots) are respectively plotted into five columns with different subsets of GAN River-I (blue dots) from avulsion rate groups one to five in Table 3.3. The red and blue dots' degree of overlap implies the level of Fluvial GAN (red) replicating the GAN River-I (blue) in model diversity. Regarding what a mode collapse case looks like in UMAP space, Figure 4.15 attaches a low diversity GAN result in the first row as the reference. This mode collapse case results from the baseline with One-Hot

Encoder while excluding the zero-centred gradient penalty term in the loss function. After training, this mode collapse case randomly simulates 1,000 realisations as the comparison set to Fluvial GAN set. When a GAN suffers mode collapse, the GAN-simulated set (the first row of Figure 4.15) references the level of data spread in the UMAP metric space. Red dots cover some regions of blue dots but are absent in many areas, particularly in the third to fifth columns in Figure 4.15, representing meandering fluvial patterns resulting from low avulsion rate settings. In contrast, the Fluvial GAN set (the second row of Figure 4.15) broadly spreads and almost reaches every place the blue dots exited in the UMAP metric space. This high overlap demonstrates Fluvial GAN can produce diverse types of fluvial patterns represented by GAN River-I, and therefore, doesn't have a serious mode collapse issue.

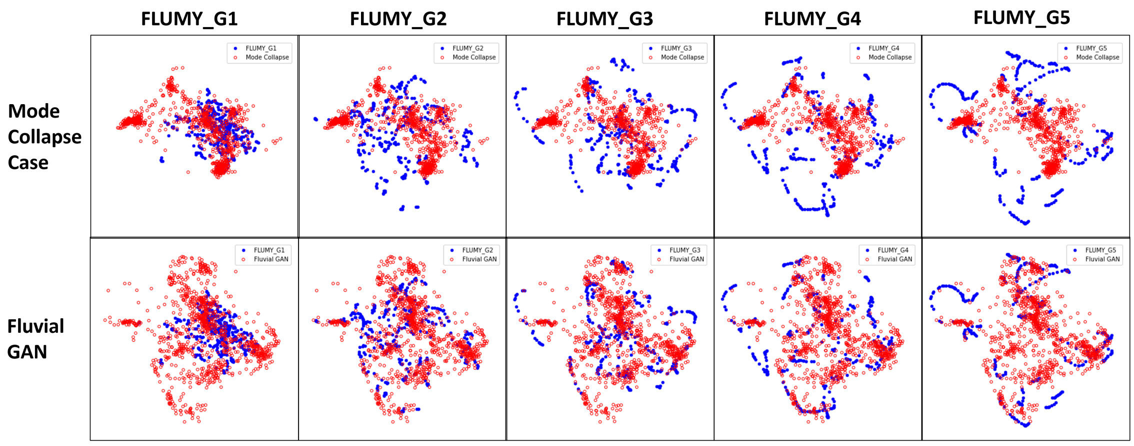


FIGURE 4.15: Diversity plot of FLUMY realisations against GANs realisations. Blue points are FLUMY realisations. Red points are GAN realisations. FLUMY_G1 to FLUMY_G5 denote FLUMY realisations with different avulsion rates shown in Table 3.3 from groups 1 to 5.

A further study analyses whether Fluvial GAN generations gathering in the same place as FLUMY realisations resemble them by visualising one FLUMY realisation and five Fluvial GAN generations nearby it in the UMAP metric space. Figure 4.16 displays an example of the visual comparison between a FLUMY realisation and its five nearest neighbours from Fluvial GAN in the UMAP metric space. Here the distance measured in the UMAP metric space is Euclidean, while the measured similarity between facies realisations is often non-Euclidean [Caers, 2011]. Therefore, the five Fluvial GAN realisations in Figure 4.16 look different from the reference realisation from GAN River-I, but they all contain a wide meander-belt in the middle of the field. The five facies models from Fluvial GAN present different fluvial patterns with diverse shapes, various sinuosities, and different channel-filled facies, representing a range of geological uncertainty.

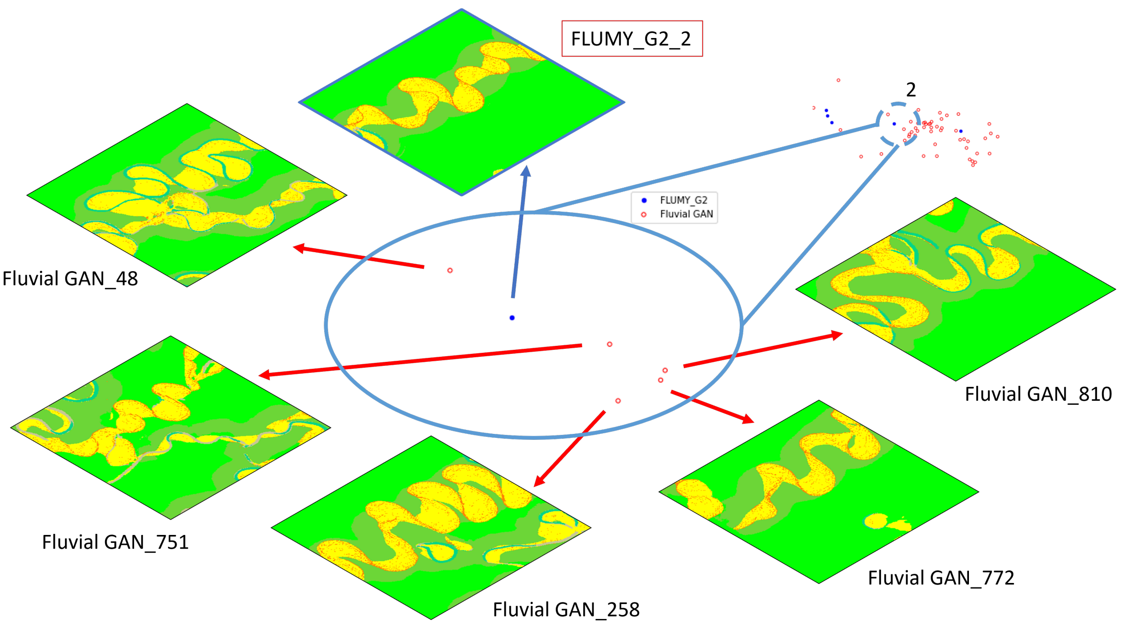


FIGURE 4.16: An example of five nearest neighbours to a FLUMY realisation in UMAP 2D space. FLUMY_G2_2 denotes a FLUMY realisation from a 3D simulation with the avulsion rate in Group 2 of Table 3.3.

The third evaluation examines if Fluvial GAN can learn the variability of fluvial patterns in a FLUMY 3D succession by searching and optimising Fluvial GAN realisations to match the slices of a given sequential succession shown in Figure 3.11. This study relies on minimising Fluvial GAN generations and the target FLUMY slice by searching and optimising the values in the latent vector of Fluvial GAN (see Figure 4.17). Fluvial GAN generations in Figure 4.17 highly resemble their FLUMY counterparts concerning the main meander belt in the middle of the field. These results illustrate Fluvial GAN can reproduce diverse meandering fluvial patterns, reflecting the river evolution modelled by FLUMY. Unfortunately, the gradient-based optimiser fails to match the thin channels because of a local minimum trap. Thus, the Fluvial GAN-simulated slices don't capture the new channel's initialisation in Figure 4.17 as it is thin.

The fourth appraisal investigates if the Fluvial GAN's latent space represents facies models properly using two random vectors for an interpolation test. This test collects the interpolated realisations by linearly changing the latent vector from vector one to vector two as the Fluvial GAN's input (see Figure 4.18). All facies models in Figure 4.18 are geologically

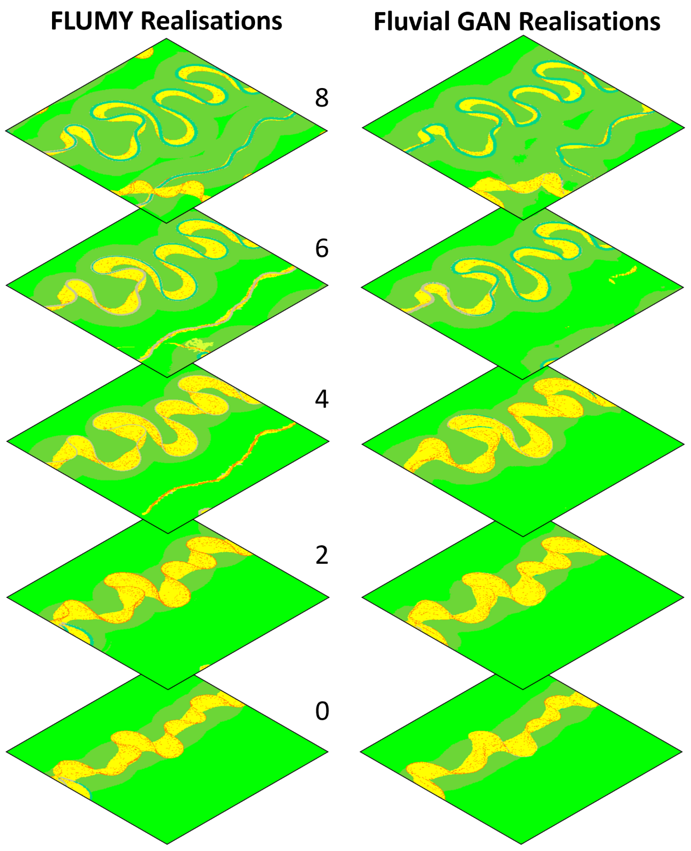


FIGURE 4.17: Optimised Fluvial GAN realisations resemble FLUMY realisations by searching at its latent vector.

plausible and resemble slices from GAN River-I, which proves the latent space of Fluvial GAN has a smooth transition. The facies model gradually shifts from less developed to mature meandering channels, indicating that Fluvial GAN successfully learns relevant representations of meandering fluvial geology [Radford et al., 2015].

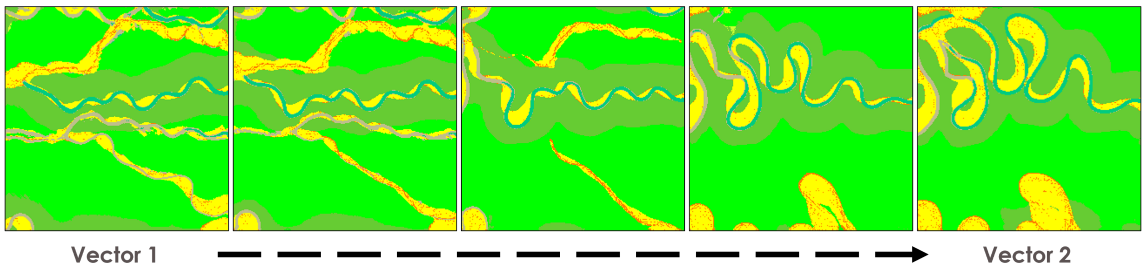


FIGURE 4.18: Interpolation between two random vectors.

According to the assessments above, Fluvial GAN learns diverse patterns of meandering fluvial geology, demonstrating GANs, with a tuned design, can handle the challenge of multi-facies meandering fluvial modelling. Fluvial GAN reproduces the fluvial channels' meandering shapes and the spatial relationship between the channels and lateral accretion packages, e.g. point bars and channel lags. Most facies models from Fluvial GAN resemble FLUMY realisations and place the levee facies properly along the meander belts.

However, Fluvial GAN still has several unsolved problems, resulting in a geologically inaccurate facies model. First, Fluvial GAN occasionally creates facies models with a clear 'gridding effect', about 2% (see Figure 4.10 d). Second, some channels in Fluvial GAN generations frequently change the filled deposits within the same channel (see Figure 4.10 e). Though this pattern may occur in nature or due to the FLUMY's discretisation program, no realisation has a single channel with a facies transition back and forth repeatedly in GAN River-I. The third issue is the kind of artefact shown in Figure 4.10 f that has a channel or meander belt's local features while detached and unconnected to any laterally consistent set of geobodies. GAN River-I contains some detached geobodies (see Figure 3.13) as each training data is a single slice of a FLUMY 3D simulation (please refer to section 3.2.3 for details). Thus, the artefact may as a result of the 2D training data containing detached geobodies shown to Fluvial GAN.

This study only tested the Fluvial GAN in reproducing low NTG meandering fluvial models from FLUMY for an easier analysis of GAN performance visually and using connectivity. The generations' quality may differ for facies models with high NTG, different fluvial systems, e.g. braided and anastomosing, or even other sedimentary environments completely. The question of how to transfer this approach to a different depositional system is beyond this work and can be seen as a further development. Such further development may require bespoke GAN optimisation for a particular geological system. Nevertheless, many original modifications, e.g. embedding One-Hot Encoder, introduced in Fluvial GAN are generic for multi-facies modelling.

Limited to the time and computing resources available, this study does not explore many other recent techniques that may help improve the GAN generation quality. For example, Zhang et al. [2019a] applied the self-attention mechanism to the convolutional neural network-based GAN, called SAGAN, helping capture long-range dependency across the input regions. Karras et al. [2019] proposed a style-based generator (StyleGAN) based on a conditional normalisation, adaptive instance normalization (AdaIN), to help with understanding and disentangling the latent vector's variation, which better separates the global features from local features (stochastic variation). The StyleGAN is trained in a progressive growing way [Karras et al., 2017], helping improve the fidelity of high-resolution image synthesis [Karras et al., 2019].

4.3 Summary of Chapter 4

This chapter presented the research route of the Fluvial GAN model for reproducing 2D realisations from GAN River-I. Commencing from a comparison study between four popular GAN variants, I decided to use the PatchGAN with a gradient-penalty term to prevent mode collapse issue as the baseline for the multi-facies modelling of meandering fluvial geology. Due to the data complexity regarding the multi-facies meandering fluvial patterns, PatchGAN realisations display some undesired features that are clearly poor reproductions of the GAN River-I, including the 'mislabelling' (noisy texture) and 'close channel' (detached loops) issues. By embedding the One-Hot Encoder into the baseline, the boundary between facies in the GAN-simulated realisations becomes visually as recognisable as its training data because the One-Hot Encoder suits to handle nominal data (categorical variable without intrinsic ordering). Then, I proposed a Hybrid-discriminator based on PatchGAN discriminator architecture and the hybrid dilation convolutional neural network, which significantly decreases the occurrence of the 'closed channel' pattern by around one magnitude. Counting the baseline's improvement on the PatchGAN's loss function, I made three enhancements to allow Fluvial GAN to learn diverse multi-facies meandering fluvial patterns from GAN River-I.

Chapter 5

Conditional GAN for Unconditional 3D Facies Modelling

This chapter proposes an iterative approach to creating 3D facies models using conditional generative adversarial networks (conditional GANs), which enables the modellers to define the reservoir thickness/the number of layers/slices. This novel conditional GAN framework learns to build 3D meandering fluvial facies models slice by slice from bottom to top, mimicking a purely aggrading depositional process. This approach conditions each new slice to the one below, making the reconstructed 3D succession geologically consistent vertically. The conditional GAN in this framework only needs to process 2D data, making it less computationally costly than the geological models used for training and 3D convolution-based GANs that use entire 3D facies models as inputs.

5.1 Challenges of Using GANs to Learn 3D Models

Training GANs to reproduce 3D geological models is still challenging, particularly for large meandering fluvial systems with complex facies transitions. Unlike GAN applications for 2D facies modelling, fewer pioneering papers focused on tackling 3D facies modelling using 3D convolutional neural network-based GANs [Laloy et al., 2018, Song et al., 2022, Yang et al., 2022, Zhang et al., 2019b]. Even though their training datasets are simpler geology than GAN River-I, the 3D convolutional neural network-based GANs still had a hard time reproducing the 3D samples, which researchers spent great efforts to tune their GAN models to a specific case [Laloy et al., 2018, Song et al., 2022, Yang et al., 2022, Zhang et al., 2019b]. This section broadly analyses the challenges of GAN for 3D reservoir facies modelling and the drawbacks of existing 3D convolutional neural network-based GAN applications.

The first challenge for GAN-based 3D reservoir facies modelling is the heavy computational burden raised by 3D training data. Previous papers applied GANs to reproduce 3D facies models by directly using the entire 3D data as the inputs in the GAN training pipeline [Laloy et al., 2018, Song et al., 2022, Zhang et al., 2019b]. The 3D data's spatial size is much more voluminous (regarding the number of pixels/elements) than the 2D images used in other 2DGAN cases, making it difficult for GAN generations to match the more complicated 3D geological patterns in the training dataset. This volume (referring to the number of pixels/voxels) may grow further if the modellers use advanced data transformation algorithms to handle multiple facies modelling. For example, Fluvial GAN uses the One-Hot Encoder to convert the facies from categories to numeric data, expanding the data volume several times that equal the number of facies. The data expansion raised by the One-Hot-Encoder further exacerbates the difficulties in pattern representation. Regarding the GAN model structure for processing 3D data, the kernel used for 3D convolution commonly has a larger spatial size than in 2D convolution since it contains three spatial dimensions instead of two. For instance, a 2D convolutional kernel whose spatial size is 3×3 has nine learnable parameters. However, when extending this kernel to do 3D convolution, its spatial size grows to $3 \times 3 \times z$, where z denotes a constant value that equals the kernel's spatial size in the z-dimension, increasing the number of learnable parameters to $9 \times z$. Thus, the 3D convolutional neural network-based GANs contain more learnable parameters, which require more CPU/GPU memory room and computation time. The gradient-based optimisation in those heavy neural networks also becomes unstable and even fails to converge [Brock et al., 2018].

One way of helping 3D convolutional neural network-based GANs to learn 3D geological patterns better is using conditioning data to provide information about facies' spatial arrangements that maintain geological consistency of vertical depositional succession. Yang et al. [2022] proposed a conditional GAN approach that reconstructs 3D facies models from 2D cross-sections. Yang et al. [2022] extracted cross-sections (2D cross-slices of facies realisations) from the 3D training dataset. Then, they use 2D cross-sections as the conditioning data to constrain GAN's generations of the 3D volume. The conditioning data helped GANs produce 3D facies models matching the geological patterns of the reference data. However, the conditional GAN model presented in Yang et al. [2022] still needs to use 3D data as the input (3D cubes with known values in the cross-sections). Therefore, the computational issues caused by the heavy model (many learnable parameters) mentioned earlier remain unsolved.

The second challenge is that it is hard to reuse a pre-trained GAN for fluvial (or other sedimentary environments) modelling to a different-sized reservoir with similar sedimentary settings unless re-training partial layers of the generator or changing the (input) latent vector size if the generator is a fully deep convolutional neural network model. Preparing the dataset and training the model are two time-consuming processes of deep learning-based applications. Thus, reusing the pre-trained model promotes the GAN-based reservoir facies modelling tools from modelling one reservoir to others with the same depositional environment. For GANs containing dense neural network layers [Chan and Elsheikh, 2019], their output size is the same as their training data. Chan and Elsheikh [2019] presented a way of reusing a pre-trained generator to simulate larger 2D images by extending its latent vector and corresponding parameters. For GANs without dense neural network layers [Laloy et al., 2018, Song et al., 2022], the output size of the generator can be different from its training data, but only by keeping a constant magnification to the latent vector in each spatial direction. Both cases limit the application of pre-trained GANs to other reservoirs with a similar sedimentary environment but different aspect ratios and sizes.

5.2 Fluvial GAN 3D

This section introduces a conditional GAN-based method for 3D reservoir facies modelling, allowing users to define the reservoir thickness and tune the 3D pattern mimicking the impact of different avulsion rates in a purely aggrading fluvial system. The two features highlighted here enable the modellers to reuse the pre-trained conditional GAN to model reservoirs with similar sedimentary settings but different thicknesses. As the conditional GAN-based method reconstructs 3D cubes layer by layer, the modellers can set the number of layers (each 1 meter thick) required in the target reservoir. Also, this approach allows the users to control the spatial correlation strength between reservoir layers, yielding different 3D geological patterns. Therefore, the modellers can adjust the types of simulated sand-body based on their understanding of the target purely aggrading meandering fluvial reservoirs' scenarios.

The reconstruction framework contains one conditional GAN-based model and two proposed training enhancements to improve the pre-trained conditional GAN's performance in reconstruction:

1) A conditional GAN-based model extends the Fluvial GAN into 3D reconstruction, named FluvialGAN_3DR, which creates upper-slices conditioned to given lower-slices using the SPADE Generator.

2) The multi-step enhancement sets up multiple schedules that train the conditional GAN to predict the slice different meters/layers above the given slice in every training iteration, preventing the predicted upper slices from gradually losing realism.

3) The conditioning data decay enhancement produces a biased lower slice as the conditioning data by summing all slices below the current target (including the base slice) in a decay mechanism.

Following the ablation study, this section first compares three cases of FluvialGAN_3DR with or without one or both proposed training enhancements above to demonstrate the two training enhancements' effects. The three cases have the same settings of model architecture but different training enhancements (see Table 5.1). Two ablation studies composed of the three cases (Case 1 VS Case 2 and Case 2 VS FluvialGAN_3DR) investigate the effect of the multi-step and conditioning data decay, respectively. After training, two ways of initiating the base slice for 3D reconstruction assess the performance of the three cases. One takes a 2D realisation from the GAN River-I dataset as the base slice for the 3D reconstruction. The other uses a pre-trained Fluvial GAN generator to randomly create 100 facies realisation as the test set to test the 3D reconstruction's performance.

The conditioning data decay enhancement introduces a hyper-parameter called the avulsion rate factor, N_{av}, heuristically correlated to the avulsion rate group labels in GAN

TABLE 5.1: Cases in the ablation study

GAN Cases	Multi-step enhancement	Conditioning data decay enhancement
Case 1	×	×
Case 2	✓	×
FluvialGAN_3DR	✓	✓

✓ denotes Yes. × denotes No.

River-I (see Table 3.3) to control the spatial correlation strength between simulated slices. To explore the effect of the N_{av} on the FluvialGAN_3DR, two further studies apply different settings of N_{av} in training and testing stages by decoupling the N_{av} and the label of the avulsion rate group in GAN River-I. The first study fixes N_{av} value of FluvialGAN_3DR to two extreme values, called the high avulsion rate variant and the low avulsion rate variant, at both training and testing stages to investigate if a constant N_{av} works as well (see Table 5.2). This experiment uses the same base slice set used to test FluvialGAN_3DR to assess the performance of the two variants. The second one varies N_{av} values of the 3D reconstruction program at the testing stage to study the impact of N_{av} on the 3D samples' geological patterns.

TABLE 5.2: Summary of the N_{av} values in different FluvialGAN_3DR cases

GAN Cases	N_{av} in train	N_{av} in test (study 1)	N_{av} in test (study 2)
High avulsion rate variant	1	1	5
Low avulsion rate variant	5	5	1
Standard FluvialGAN_3DR	[1,5]	[1,5]	1 and 5

This study uses both quantitative and qualitative methods to analyse the results, as quantitative measures are less subjective, but qualitative interpretation often conveys information about highly complex patterns that are hard to quantify. The qualitative analysis relies on visual interpretations of the GAN-simulated results by slicing horizontal and vertical sections. The quantitative indicators are sand proportion and connectivity in 3D. As the NTG of the training data is about 0.2, the 3D sand connectivity bears a big uncertainty. If Fluvial GAN 3D successfully captures the sand proportion range of the training dataset, the connectivity uncertainty in Fluvial GAN 3D generations can be another indicator for performance evaluation by comparing the cascade zones between the training dataset and GAN-simulated ensembles. The cascade zone quantifies how well the GAN reconstructed

3D facies models ensemble represents geological uncertainty across the ensemble subject to sedimentary uncertainty in the avulsion rate.

5.2.1 Fluvial GAN 3D Reconstruction Framework

The FluvialGAN_3DR's training workflow uses a similar configuration to Fluvial GAN while containing an extra data pre-processing, replacing the deep convolutional generator with the SPADE generator (please refer to Section 2.3.3.3 for details of SPADE and SPADE generator) to generate conditional realisations and changing the centre value of the gradient penalty from zero to one (see Figure 5.1). As for the use of the centre values in the gradient penalty calculation, please refer to Section 4.1.1 for the details.

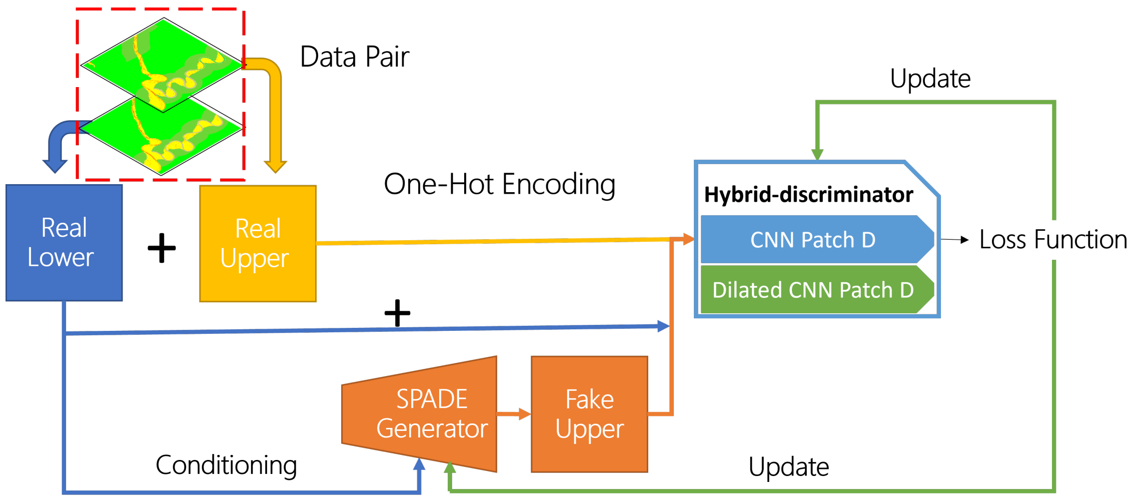


FIGURE 5.1: The workflow of training Fluvial GAN 3D reconstruction model. Single pair refers to the pair data containing a 2D realisation and the realisation one meter above it. The plus sign denotes vertically concatenating the two 2D realisations before feeding into the discriminator. One-Hot Encoding refers to using the One-Hot Encoder to pre-process data. CNN is short for convolutional neural networks. Patch D refers to the PatchGAN discriminator. The loss function is an improved version of hinge loss by adding the one-centred gradient penalty term. One-Hot Encoder, Hybrid-discriminator and the improved loss function are all from Fluvial GAN.

Regarding the data pre-processing, FluvialGAN_3DR needs to use data pairs that each training data pair contains two overlaying 2D horizontal slices throughout the reservoir instead of single images as the input for the discriminator. The two slices in each data pair are the succeeding layers, which one comes above the other with a one-meter interval in this study. In the data pair, the lower slice is the conditioning data, and the upper slice is the target conditioned on the slice below. Later implementations with the conditioning data decay enhancement compute the conditioning data differently but also use the data pair by concatenating the conditioning data and the prediction target as the discriminator's input. For short, this study calls the conditioning data the 'lower slice' and the target 'upper slice' at each training or testing iteration. FluvialGAN_3DR takes pair data as the discriminator's input because all 2D slices from the GAN River-I dataset can present as the lower slice and upper slice in the data pair during training. If the discriminator uses single images as the input, the generator will only duplicate the lower-slice data given as the conditioning data as its generation output to fool the discriminator successfully, failing the conditional GAN to predict the target (the upper-slice data).

The data pre-processing module pairs every two 2D slices, one 2D slice and the slice one meter above it, whose index value difference is ten, from GAN River-I into a data pair format as the FluvialGAN_3DR's training dataset. GAN River-I contains 16,000 slices from 25 3D facies realisations with 64 meters thick (640 0.1 meters thick layers) covering a range of uncertainty across the change of avulsion rate. While this training dataset volume is 14250 training data in the format of data pairs because the multi-step training enhancement (see more details in Section 5.2.2) requires more slices further above the base slice (7 meters/70 index values), resulting in the reduction of usable slices in the GAN River-I dataset.

FluvialGAN_3DR's model architecture is similar to the Fluvial GAN. The architectures of FluvialGAN_3DR's generator, SPADE generator, and discriminator, Hybrid discriminator (composed of two PatchGAN-based discriminators), are summarised in Table 5.3, Table 4.4 and 4.5, respectively.

The training process runs 50 epochs using the ADAM optimiser [Kingma and Ba, 2014] to update the generator and the discriminator in turn. The learning rate is 0.0002, and the values of betas are 0 and 0.9, respectively, in the ADAM optimiser.

Reproducing the meandering facies realisations in GAN River-I requires a pre-trained Fluvial GAN generator and a pre-trained FluvialGAN_3DR SPADE generator. The reconstruction program initialises a base slice as the pre-trained SPADE generator's conditioning data to simulate its upper slice and then replaces it with the simulated upper slice as the conditioning data for the next upper slice's prediction till reaching the required number of

Table 5.3: SPADE Generator Architecture

Layer	State size
Linear (128, 32768)	(Batch size, 32768)
Reshape	(Batch size, 512, 8, 8)
SPADEResBlk, Up sample (2)	(Batch size, 512, 16, 16)
SPADEResBlk, Up sample (2)	(Batch size, 256, 32, 32)
SPADEResBlk, Up sample (2)	(Batch size, 128, 64, 64)
SPADEResBlk, Up sample (2)	(Batch size, 64, 128, 128)
SPADEResBlk, Up sample (2)	(Batch size, 32, 256, 256)
SPADEResBlk, 3X3Conv-7, SoftMax	(Batch size, 7, 256, 256)

3X3Conv-7 denotes 3X3 convolutional layer with seven filters

slices/meters (see Figure 5.2). The initial base slice of this reconstruction program can be alternatively from a pre-trained Fluvial GAN or an arbitrary 2D training data from the GAN River-I dataset. Then the users need to assign values for two parameters in this reconstruction program: the first one is the number of slices reconstructed, which is essential to gain the flexibility for the conditional GAN to reconstruct the reservoir of arbitrary thickness, and the second one, named the avulsion rate factor N_{av}, which is optional but more related to the geological uncertainty associated with avulsion, biasing the correlation strength between the current target and earlier simulated slices below it. The following sections will introduce those two parameters in detail.

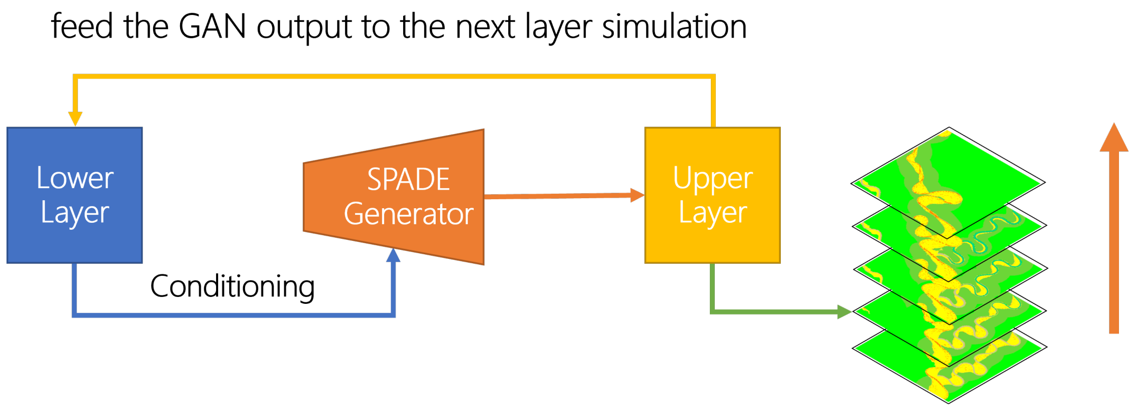


Figure 5.2: The workflow of 3D reconstruction process after training. The Lower Layer refers to the conditioning lower-slice data used in the SPADE generator. The Upper Layer is the generated upper slice from the pre-trained SPADE generator.

5.2.2 Multi-step Training Enhancement

The multi-step training enhancement incorporates the reconstruction process into the conditional GAN training by setting up multiple prediction targets (steps) at each training iteration. This enhancement has two hyper-parameters, the reconstruction step and the reconstruction iteration in each step. The reconstruction step determines the number/times of reconstructions carried out at each training iteration of the training dataset looping. Each reconstruction defines a reconstruction iteration that controls how many slices above the base slice will be calculated iteratively. The reconstruction process during training remains the same in the testing stage, which uses the conditional generator to predict the upper slice conditioned on a given lower slice and then takes the generated upper slice as the new conditioning data for the next upper slice's prediction. Each step updates the generator and the discriminator in turn. As the input data pair for the discriminator consists of the conditioning data and the prediction target, the fake and real data pairs have different lower slices (conditioning data) except for the first step, where reconstruction iteration is one (see Figure 5.3). In each step, the SPADE generator keeps using the upper slice as the new lower slice to create the next upper slice until it reaches the number of reconstruction slices required. The latest lower (conditioning data) and upper (prediction target) slices comprise the fake data pair as the discriminator's input during training shown in Figure 5.1. Regarding the real data pair, the data pre-processing module must fetch the corresponding lower and upper slices from the training dataset according to the value of reconstruction iterations at the current reconstruction step. The rest are the same as the previous framework shown in Figure 5.1.

The multi-step training framework opens a further question of optimising its hyper-parameters: the number of steps and the reconstruction iterations at each step. This study achieved a stable version based on a trial and error and left the theoretical hyper-parameter optimisation for future studies. The number of steps is four in this training program, which processes four times in each iteration for each batch of the data with reinitialised input vectors. The reconstruction iterations (the number of slices reconstructed) in the four steps sequentially are one, three, five and seven.

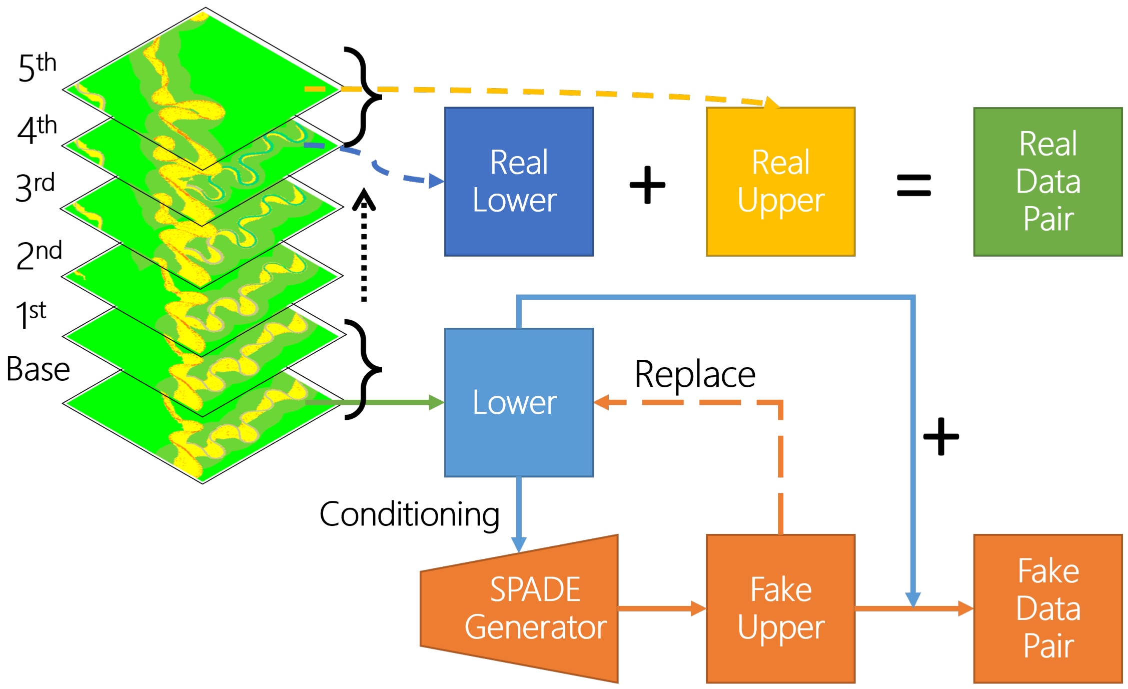


FIGURE 5.3: A schematic diagram of the multi-step training enhancement. Any two neighbouring slices with a one-meter interval from the training data can compose a real data pair.

The first ablation study compares Cases 1 and 2 by picking the same slices from the GAN River-I dataset as the base slice for the reconstruction. Initially, Case 1 and Case 2 can produce plausible upper slices. However, The upper slices produced by Case 1 start gradually losing geological realism with the reconstruction continuing to build slices upwards (see Figure 5.4 a). The reconstructed slices barely present a big chunk of sand-bodies (point bars) mixed with broken sand and mud plugs. In contrast, reconstructed realisations from Case 2 still show plausible meandering channels and appropriate point bars placements at the channels' inner banks, though the overall image quality is visually not as good as GAN River-I or Fluvial GAN's realisations (see Figure 5.4 b).

This ablation study proves that the multi-step enhancement significantly improves the performance of the conditional GAN framework. Based on visual interpretations, the upper slices (above the 10th) generated by Case 2 are obviously more plausible regarding the geological realism than those from Case 1, which doesn't need quantitative scores to verify this argument. To freely define the reservoir thickness (number of layers) when building the 3D facies models, preserving geological realism is essential in the reconstruction process, which warrants using the multi-step training enhancement.

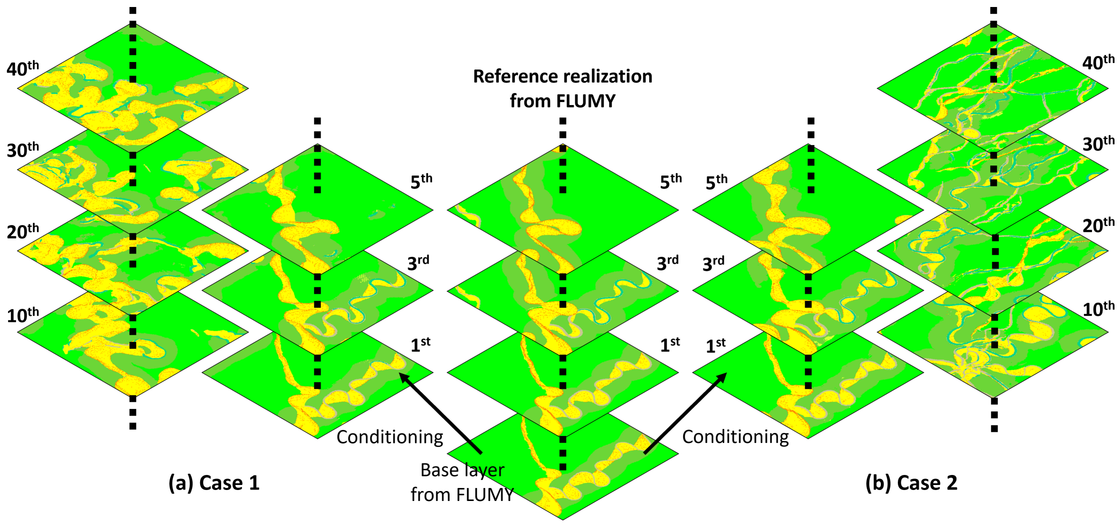


FIGURE 5.4: Comparison of meandering patterns between Case 1 and Case 2. (a) is Case 1, and (b) is Case 2.

Compared to Case 1, Case 2 implicitly involves the generation history in predicting the new upper slice. Case 1 fully ignores the impact of the generation history and only uses the latest generation to predict the next upper slice. This leads Case 1 to forget history, exactly what the assumption of the Markovian process is that X_{t+1} depends on X_t [Gagniuc, 2017]. Instead, Case 2 uses the discriminator to penalise the generated upper slices in pair that are further above the base slice. In this way, Case 2 incorporates the history into the 3D reconstruction sequentially throughout the conditional generator. However, the conditional GAN in Case 2 is not trained in a recurrent way, which means the current generation and the input end (SPADE's input) of the next generation are directly connected. Instead, the current generation is disconnected by detaching the gradients before using the current generation as the SPADE's input for the next generation. This choice is because of the GPU memory limitation. Thus, Case 2 doesn't explicitly involve the generation history in its prediction, as it still only uses the latest generation as the conditioning data.

5.2.3 Conditioning Data Decay Training Enhancement

The second conditional GAN training enhancement adds earlier simulated slices (generation history) into the conditioning data (the lower slices) *with a decay mechanism* to predict the next upper slice (target), named the conditioning data decay enhancement. This decay mechanism mimics the gradient-based optimisers with the moment mechanism containing a decay parameter [Kingma and Ba, 2014, Qian, 1999]. The real data pair's lower slice is an exponentially decayed mean of slices between the base slice and the latest lower slice, where those slices are continuous values and no longer treated as one-hot encoded indicator variables. The lower slice is the conditioning data instead of the facies indicator map. This lower slice is calculated by Equation 5.1

$$c = \sum_{i=1}^{N_{iter}-1} 0.1N_{av} \cdot c + (1 - 0.1) \cdot N_{av} \cdot x_i \tag{5.1}$$

where c is the lower layer, initialised as x_0 (base layer). x refers to the real data, i denotes the layer index of real data, N_{iter} is the number of total reconstruction iterations at the current reconstruction step, and N_{av} is the avulsion rate factor. The *avulsion rate factor*, N_{av}, equals the avulsion rate group number in GAN River-I, heuristically, to impact the correlation strength between slices in the standard configuration of FluvialGAN_3DR. Therefore, the value of N_{av} is between 1 and 5; a smaller value means a higher avulsion rate and vice verse. By coupling the N_{av} value to the avulsion rate group, the decay mechanism in this training enhancement imitates the spatial correlation between layers in a meandering fluvial reservoir. A high avulsion rate leads to the river moving its channel quickly to a different zone of the modelled field. Thus, one layer becomes less likely to have a strong spatial correlation to those layers further below it if the reservoir results from a high avulsion rate purely aggrading depositional setting.

Calculating the fake pair must involve the conditioning data decay mechanism in the multi-step training framework. The algorithm 1 presents the pseudo-code of implementing the conditioning data decay enhancement in the multi-step training framework. The repeated times t denotes to the total reconstruction iterations at the current reconstruction step, and the input vector z remains the same value within each reconstruction step for simplicity. Indeed, the input vector can vary during training or testing, like as the 'seed' value in a stochastic process, resulting in different realisations. However, the question about the impact of changing the input vector of the conditional generator in reconstruction is beyond this study's scope. Thus, all cases in this chapter don't change the value of the input vector for both the pre-trained Fluvial GAN generator and SPADE-based conditional generator when simulating a single 3D facies model.

Algorithm 1: Multi-step training enhancement with conditioning data decay

```
Input: input vector $z$, repeated times $t$, avulsion rate factor $N_{av}$, the generator $G(\cdot)$,
base slice $x_0$

if $t = 1$ then
    $\tilde{c} \leftarrow x_0$;
    $\tilde{x_t} \leftarrow G(z, \tilde{c})$;
else
    $\tilde{c} \leftarrow x_0$;
    $\tilde{x_t} \leftarrow G(z, \tilde{c})$;
    for $i = 1, ..., t$ do
        $\tilde{c} \leftarrow 0.1N_{av} \cdot \tilde{c} + (1 - 0.1) \cdot N_{av} \cdot \tilde{x_t}$;
        $\tilde{x_t} \leftarrow G(z, \tilde{c})$;
    end for
end if

Output: synthesis data $\tilde{x_t}$, accumulated synthesis condition $\tilde{c}$
```

This second ablation study quantitatively compares the performance of Case 2 and FluvialGAN_3DR to investigate the effect of the conditioning data decay enhancement by creating 100 3D realisations with 32 meters/slices (1 base slice and 31 reconstructed slices). A pre-trained Fluvial GAN generator simulates a test set of base slices and connects to the reconstruction program to work as the 3D facies model simulator, Fluvial GAN 3D. This study uses the same set of input vectors for the two 3D facies model simulators (Case 2-based and FluvialGAN_3DR-based) to avoid introducing bias from different input vector initialisation. The avulsion rate group labels distribute evenly in the training dataset, and therefore, FluvialGAN_3DR assigns N_{av} values to 1 to 5 evenly during training. So, this study changes N_{av} values every 20 samples commencing from 1 to 5 in the reconstruction process (testing stage) of FluvialGAN_3DR. The following two sections, section 5.2.4 and 5.2.5, will further investigate the effect of the avulsion rate factor. The sand proportion and connectivity are the two quantitative measures to plot GAN realisations versus reference realisations from GAN River-I in this study.

The 3D facies realisations from FluvialGAN_3DR cover ranges of sand proportion and connectivity much wider than the ensemble from Case 2. This comparative result demonstrates the conditioning data decay enhancement significantly improves the conditional GAN-based 3D reconstruction's performance (see Figure 5.5). The FluvialGAN_3DR generations show both low and high connectivity, covering the uncertainty in the GAN-River-I dataset. Given the same initialisation, realisations from Case 2 are mainly high connectivity models. The Case 2 plot's Cascade zone is noticeably smaller, illustrating it fails to cover the connectivity uncertainty represented by the GAN-River-I dataset. Thus, with the assistance of the conditioning data decay enhancement, the SPADE-based conditional generator learns better than without it.

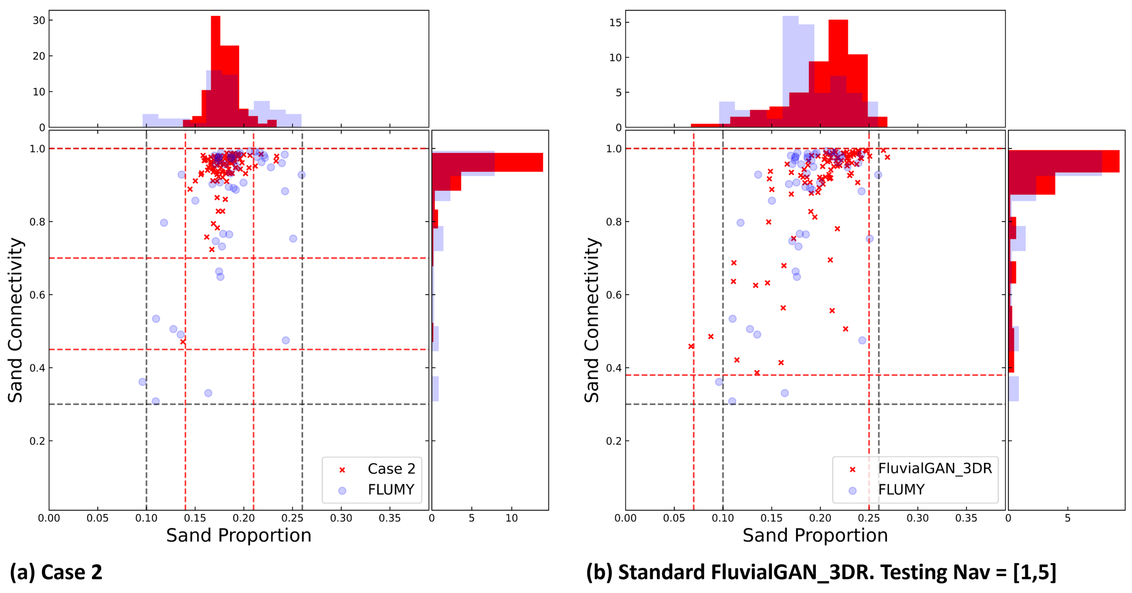


FIGURE 5.5: Comparison of sand connectivity against proportion plots between Case 2 and FluvialGAN_3DR. The N_{av} value of the FluvialGAN_3DR ranges from 1 to 5 during training. (a) is Case 2, and (b) is the standard FluvialGAN_3DR with varying N_{av} during the reconstruction process. The blue dots are the reference samples from FLUMY. The red cross markers are samples from GANs.

A further analysis studies what fluvial features were not captured in Case 2 and evaluates how this influences the resulting range of connectivity. The connectivity plot against avulsion rate group labels reveals that the low avulsion rate settings are more likely to result in low connectivity models (see Figure 3.6 b). As the sand proportion is similar, the river has more time to develop its channel laterally, leading to sand-bodies growing wider at the inner banks in a low avulsion rate depositional environment. Once avulsion happened, the sand starts accreting at different places in a distance, making the sand bodies less likely to connect with each other. In contrast, a high avulsion rate sedimentary environment promotes the river to shift its channel frequently and distribute the elongated sand bodies broadly in the modelled field, rising the opportunity for sandbodies to connect to each other.

A visual check of all horizontal slices from the bottom to the top reveals that nearly all upper slices from Case 2 present less-developed channels with elongated sand-body shapes, consistent with the connectivity VS avulsion rate group labels relationship found in GAN River-I. The vertical sections of the 3D realisations from Case 2 further prove that the conditional GAN tends to produce slices reflecting high avulsion rates regardless of using which type of realisation as the base slice (see Figure 5.6). Figure 5.6 presents an example of Case 2 generations inconsistent with the preset sedimentary setting, changing the type of sand-bodies from sheet-type to ribbon-type. Case 2's SPADE-based generator tends to abandon and re-initialise a channel (avulsion) instead of developing it, indicating that the conditional GAN seems to easier have a lower cost/penalty in this way.

As discussed in section 5.2.2, the multi-step enhancement incorporates previously simulated upper slices at earlier steps in training but only uses the latest simulated upper slice (the slice directly below the current target) as the conditioning data in the conditional GAN-based framework. In other words, the newest-generated realisation is not directly correlated to previously simulated slices except for the one directly below it. This correlation works fine for high avulsion rate depositional environments but doesn't represent the reality in low avulsion rate depositional environments. In a low avulsion rate purely aggrading depositional environment, a 3D model's horizontal layers have stronger spatial correlations vertically as the river is less frequently changing its channel to another place far away from the current position. Therefore, lower avulsion rates increase the possibility that one horizontal layer has a channel located nearby the same place in the layers below it. Lacking spatial correlations to layers further below the target accounts for the 3D realisations from Case 2 missing well-developed meandering fluvial patterns in their upper slices.

To address the correlations between horizontal slices, FluvialGAN_3DR incorporates previously simulated upper slices in the conditioning data with a decay mechanism, mimicking the reality that closer slices have stronger spatial correlations. The correlation strength is heuristically correlated to the avulsion rate of fluvial systems by a *avulsion rate factor*, N_{av}, which impacts the calculation in the conditioning data decay enhancement. Given the same input vector and, therefore, the same base slice shown in Figure 5.6, FluvialGAN_3DR develops meandering channels, favouring more lateral-extended (sheet-type)

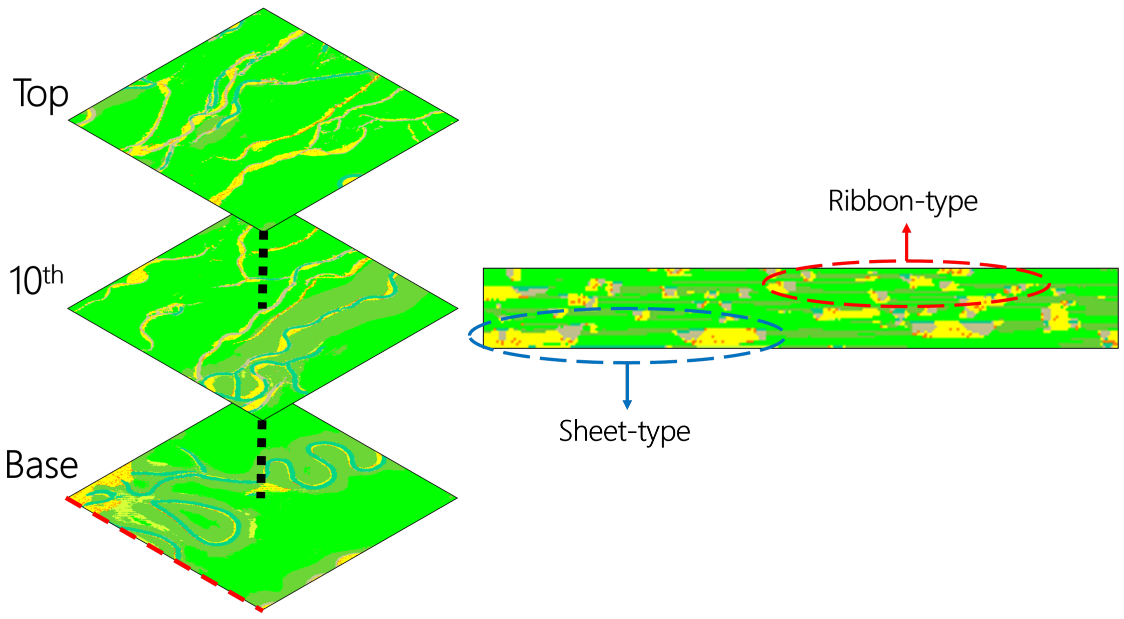


FIGURE 5.6: Horizontal and vertical slices of an example initialised with a developed meandering pattern from Case 2.

sand bodies (see Figure 5.7). Please be noticed that the N_{av} value is 5 when reconstructing this sample, and please refer to Section 5.2.5 for the effect of N_{av} on FluvialGAN_3DR.

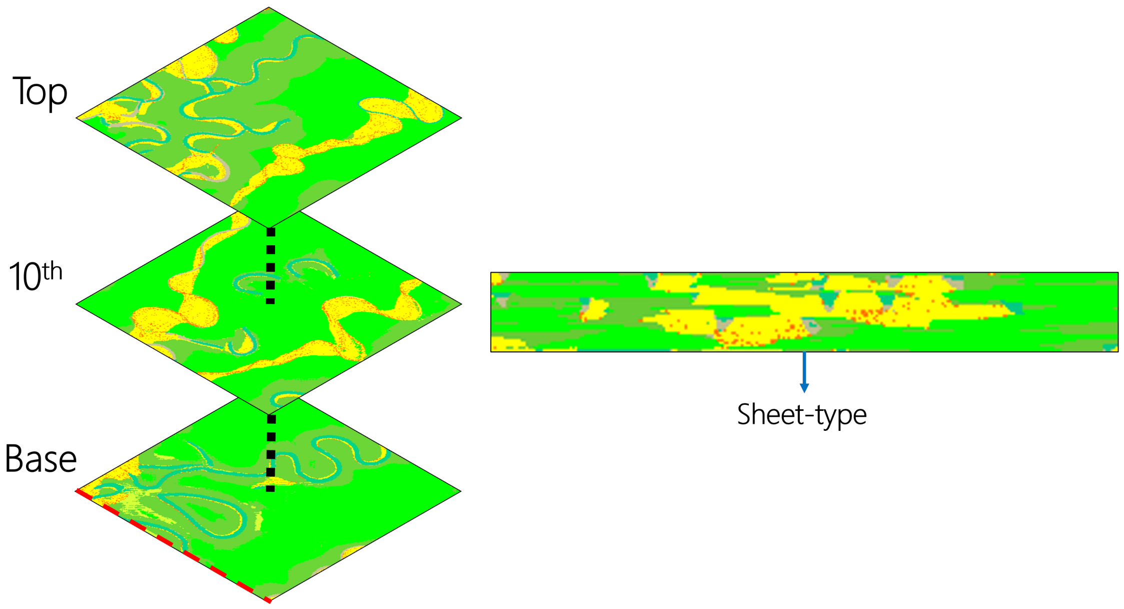


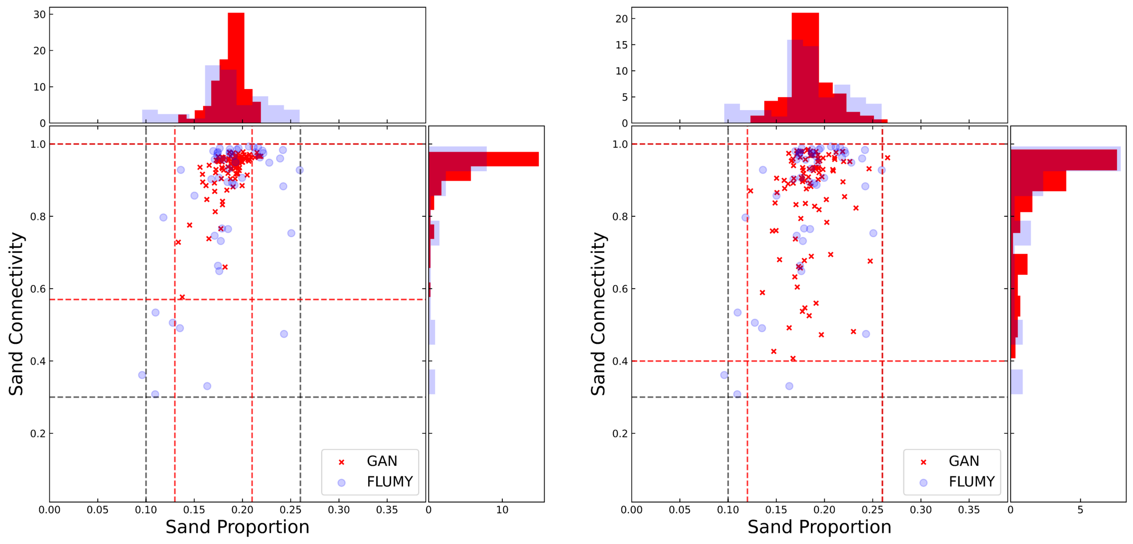
FIGURE 5.7: Horizontal and vertical slices of an example initialised with a developed meandering pattern from FluvialGAN_3DR.

Compared to Case 2, FluvialGAN_3DR explicitly incorporates generation history into its prediction by adding the historical generations to conditioning data with a decaying mechanism. This enhancement shares the same idea as the autoregressive process that uses historical values to predict the next value of the output [Box et al., 2015]. For both classical and deep autoregressive models, they predict a single sequence/image value by value [Box et al., 2015, Hoogeboom et al., 2021, Menick and Kalchbrenner, 2018, Van den Oord et al., 2016]. Instead, FluvialGAN_3DR directly creates the 2D facies model by using historical generations as the conditioning data. In other words, FluvialGAN_3DR can be regarded as training a conditional GAN model to generate a 3D cube slice-by-slice in an autoregressive way.

5.2.4 The Effect of Avulsion Rate Factor during Training

This section investigates the effect of the avulsion rate factor, N_{av}, on training the conditional GAN-based framework to verify if decoupling the decay mechanism and avulsion rate is appropriate. The standard FluvialGAN_3DR has varied N_{av} values correlated to the avulsion rate. This study builds two variants of FluvialGAN_3DR by fixing their N_{av} values to two extreme constants: the high avulsion rate variant ($N_{av} = 1$) and the low avulsion rate variant ($N_{av} = 5$). The high and low avulsion rate variants have the same model architecture and training configurations. After training, both variants reconstruct 100 3D realisations using the same sets of base slices by feeding the same input vectors. Then, the results of the two sets of 3D samples show in a plot based on their sand proportion and connectivity (see Figure 5.8). Compared to the standard configuration of the FluvialGAN_3DR, the high avulsion rate variant shows a smaller Cascade zone in the NTG-connectivity plot. On the other hand, the low avulsion rate variant appears to present a wider Cascade zone and a similar data cloud spread.

Visual interpretations of the reconstructed 3D realisations from the high and low avulsion rate variants reveal the geological reason for the different behaviours in connectivity. The high avulsion rate variant is similar to Case 2, preferring abandon and re-initialise channels with the reconstruction upwards (see Figure 5.9 a). On the other hand, the low avulsion rate variant can keep developing meandering fluvial realisations given to it while maintaining the type of geological patterns represented by the base slice (see Figure 5.9 b). The spatial correlation strength variance caused by different N_{av} values accounts for the difference in geological patterns at reconstructed upper slices. Based on Equation 5.1 and Algorithm 1, a smaller N_{av} value assigns the biggest weight to the slice immediately below, with the weight descending for deeper slices. Thus, the latest generation (the slice immediately

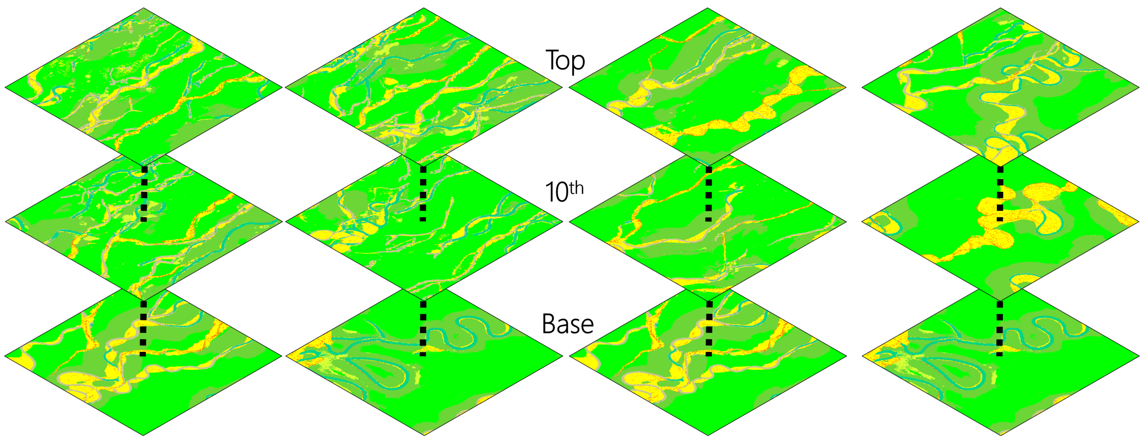


(a) The high avulsion rate variant. Training Nav = 1; Testing Nav = 1.

(b) The low avulsion rate variant. Training Nav = 5; Testing Nav = 5.

FIGURE 5.8: Comparison of sand connectivity against proportion plots between the high and low avulsion rate variants. (a) is the high avulsion rate variant. (b) is the low avulsion rate variant. The blue dots are the reference samples from FLUMY. The red cross markers are samples from GANs.

below the current prediction target) greatly impacts the prediction of the high avulsion rate variant. In contrast, previously simulated slices (deeper slices) decay rapidly, resulting in a limited impact on the high avulsion rate variant's prediction.



(a) The high avulsion rate variant.

(b) The low avulsion rate variant.

FIGURE 5.9: Comparison of meandering patterns between the high and low avulsion rate variants. (a) is the high avulsion rate variant ($N_{av} = 1$ during training). (b) is the low avulsion rate variant ($N_{av} = 5$ during training).

According to the analyses above, selecting a big N_{av} value does not undermine the 3D reconstruction performance in connectivity uncertainty coverage and geological pattern consistency. However, the N_{av} value in FluvialGAN_3DR training clearly has a significant impact on the reconstruction performance at the testing stage. This result opens a further question: what will happen if only varying the N_{av} in the reconstruction program at the testing stage?

5.2.5 The Effect of Varying Avulsion Rate Factor during Reconstruction

A further study explores the effect of using different values of the avulsion rate factor, N_{av}, in the reconstruction process at the testing stage to investigate if it is a potential way of blending the modellers' understanding of the avulsion rate settings into GANs' generations. This study uses the standard configuration and the two variants of FluvialGAN_3DR presented in Section 5.2.4 to evaluate their performance with different N_{av} values at the testing stage. All three pre-trained FluvialGAN_3DR generators reconstruct the same set of base slices for two rounds by setting N_{av} values to one and five, respectively, in the reconstruction process. Then, both quantitative and qualitative analyses compare the reconstructed 3D realisation sets by calculating each sample's sand proportion and connectivity and visualising all slices within this sample.

Firstly, changing N_{av} to the opposite end value in the reconstruction process for both low and high avulsion rate variants leads to worse performance. For the high avulsion rate variant ($N_{av} = 1$ in the training process), setting N_{av} to 5 in the reconstruction process results in generating a big area of background (overbank facies) after reconstructing about 10-20 slices, which causes the data point cloud (the ensemble of realisations) shifts to the left side in the proportion-connectivity plot (see Figure 5.10). For the low avulsion rate variant ($N_{av} = 5$ in the training process), setting its N_{av} to 1 in the reconstruction process at the testing stage doesn't cause the blank slices issue, which still creates plausible meandering fluvial channels (see Figure 5.11). The data point cloud reasonably shrinks to a narrower range in the proportion-connectivity plot. However, the overall quality gets worse based on visual interpretation. More importantly, when the base slice presents less developed meandering channels that can either be abandoned or developed later, the low avulsion rate variant prefers avulsion regardless of the N_{av} values in the reconstruction process at the testing stage (See Figure 5.9 b and 5.11).

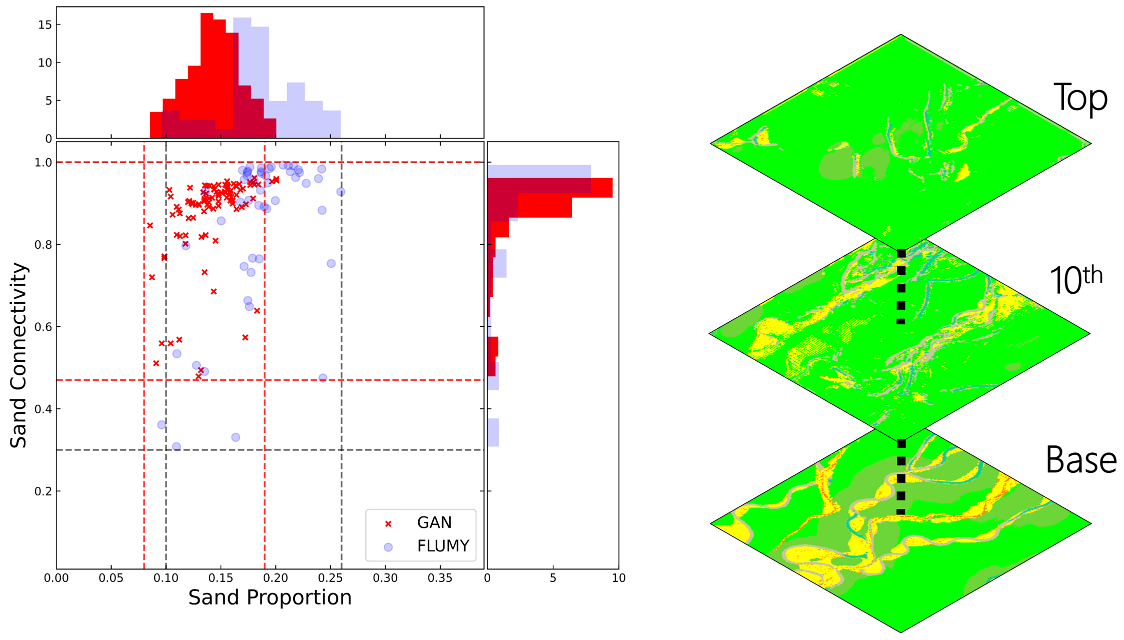


Figure 5.10: Sand connectivity against proportion plot and slices visualisation of the high avulsion rate variant ($N_{av} = 1$ during training) when setting N_{av} to 5 in the reconstruction process.

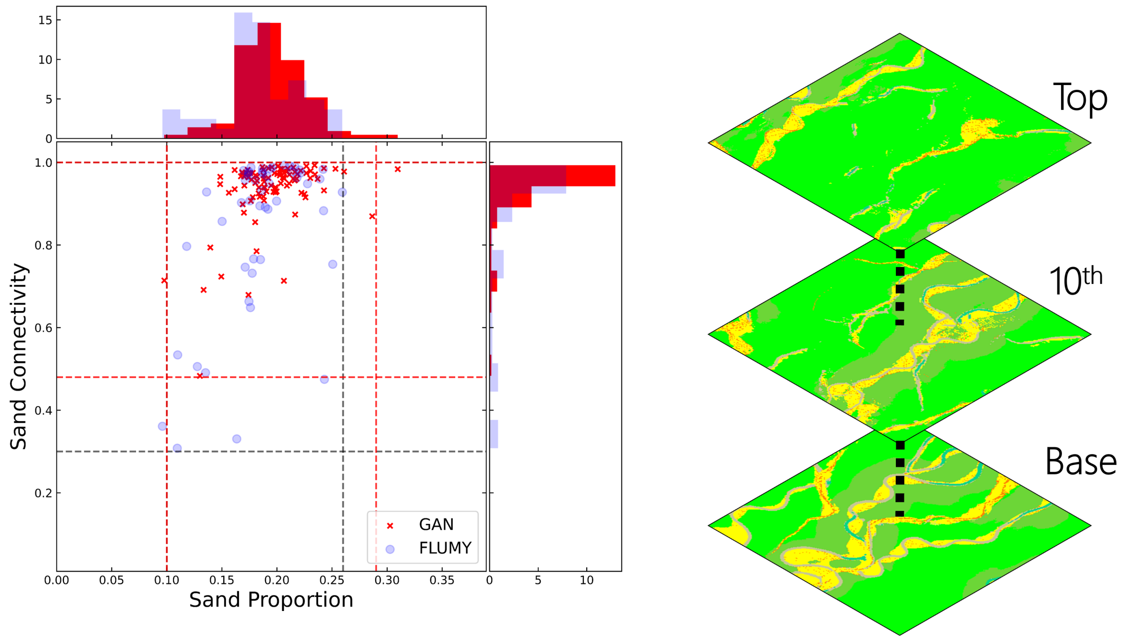
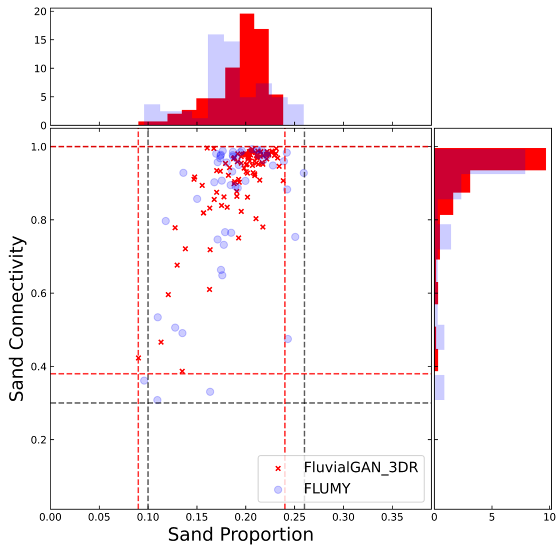
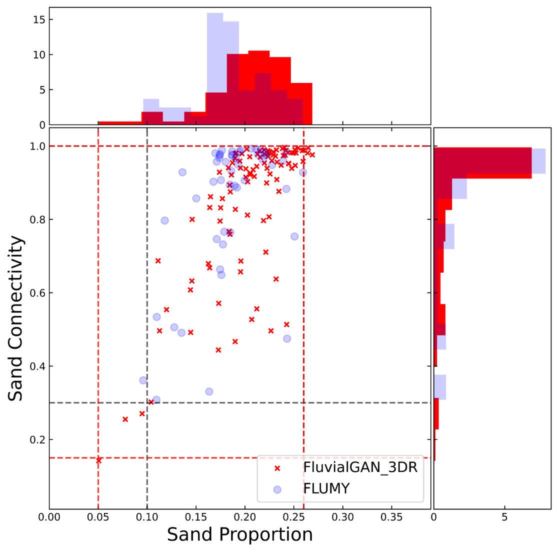


Figure 5.11: Sand connectivity against proportion plot and slices visualisation of the low avulsion rate variant ($N_{av} = 5$ during training) when setting N_{av} to 1 in the reconstruction process.

Then, the standard FluvialGAN_3DR reconstructs the same set of base slices using the two extreme constants (1 and 5), presenting a clear shift in the proportion-connectivity plot when varying the N_{av} value from one to five (see Figure 5.12). When N_{av} equals 1 in the testing reconstruction, the 3D realisation ensemble from FluvialGAN_3DR shows slightly narrower ranges of sand proportion and connectivity than GAN River-I. In contrast, the 3D realisation ensemble from FluvialGAN_3DR covers wider ranges of sand proportion and connectivity than GAN River-I when N_{av} is 5 in the testing reconstruction. The 3D realisations from GAN River-I result from different avulsion rate settings that have a direct correlation to the connectivity in this study (see Figure 3.6) – low avulsion rate models show a bigger connectivity uncertainty. Therefore, the variance of ranges shown in Figure 5.12 is reasonable as a higher N_{av} value indicates a lower avulsion rate. Due to the stochastic nature, the base slices set generated by a pre-trained Fluvial GAN contains diverse meandering fluvial patterns reflecting various sedimentary settings regarding avulsion rates at different channel evolution stages. Thus, even if the N_{av} value is 1 (representing high avulsion rate settings) during testing reconstruction, the reconstructed 3D realisations can still show low connectivity.



(a) Standard FluvialGAN_3DR. Testing Nav = 1



(b) Standard FluvialGAN_3DR. Testing Nav = 5

FIGURE 5.12: Comparison of sand connectivity against proportion plots when changing N_{av} from 1 to 5 in FluvialGAN_3DR reconstruction process. (a) is setting N_{av} to 1 during reconstruction. (b) is setting N_{av} to 5 during reconstruction. The blue dots are the reference samples from FLUMY. The red cross markers are samples from GANs

A further visual interpretation illustrates varying the avulsion rate factor is a possible way of incorporating the modellers' understanding into FluvialGAN_3DR's reconstruction.

FluvialGAN_3DR can create different types of meandering fluvial patterns by changing the N_{av} value when the base slice shows a less developed meandering pattern (same base slice as in Figure 5.10 and 5.11). The standard FluvialGAN_3DR creates more elongated channels when setting N_{av} to one in the reconstruction process (see Figure 5.13 a). Given the same base slice, the standard FluvialGAN_3DR produces a more meandering channel when switching N_{av} to five in the reconstruction process (see Figure 5.13 b). This difference in geological patterns reflects the different geological settings and proves that the modellers can modulate the produced 3D realisations based on their understanding of the vertical correlation strength in their reservoirs.

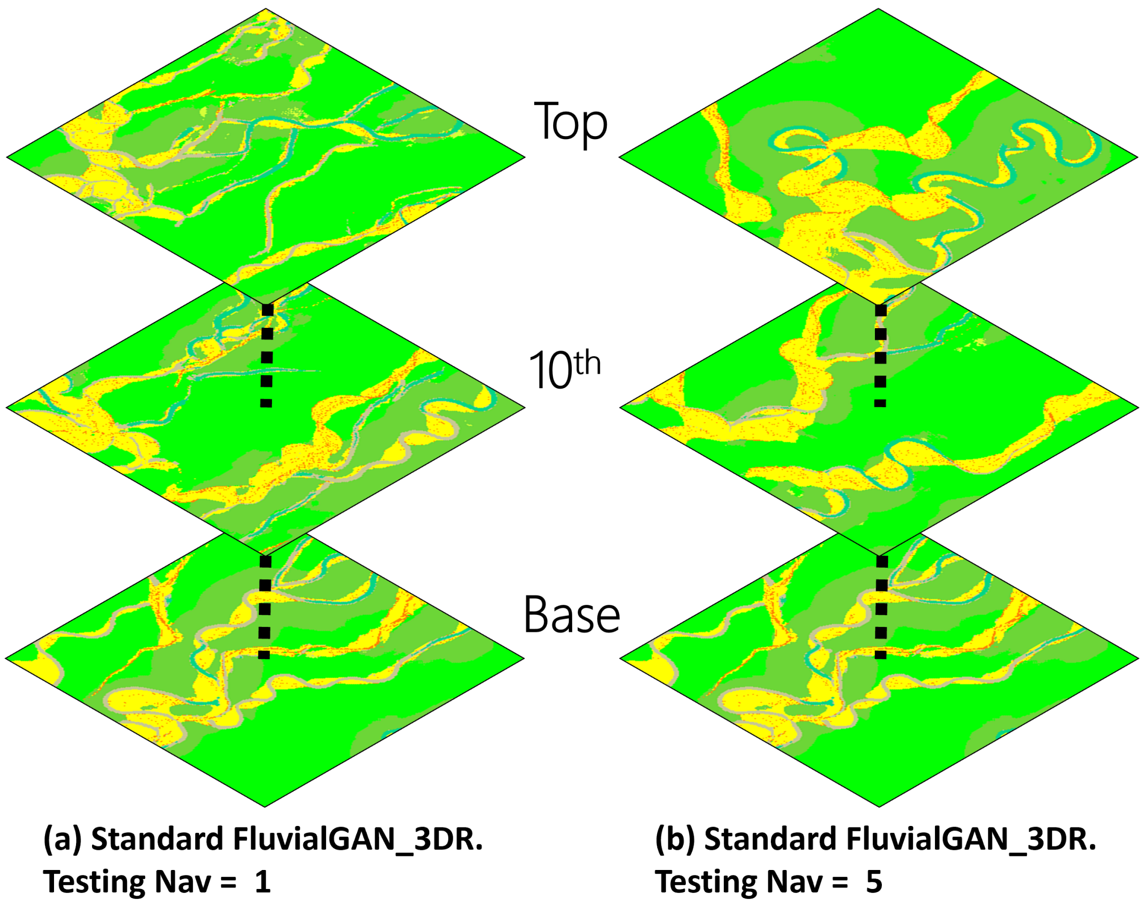

Top
10th
Base
(a) Standard FluvialGAN_3DR. Testing Nav = 1
(b) Standard FluvialGAN_3DR. Testing Nav = 5

FIGURE 5.13: Comparison of meandering patterns when changing N_{av} from 1 to 5 in the standard FluvialGAN_3DR reconstruction process. The N_{av} value of the standard FluvialGAN_3DR ranges from 1 to 5 during training. (a) is setting N_{av} to 1 during reconstruction. (b) is setting N_{av} to 5 during reconstruction.

However, the conditioning data decay enhancement of FluvialGAN_3DR is subject to an assumption of a purely aggrading system and, therefore, cannot reflect the occurrence of incisions. First, the GAN River-I dataset used in this study is 3D facies models simulated without incision. Thus, no data is available to verify whether the proposed approach works in an incision case. Second, the conditioning data decay enhancement assumes a correlation between the avulsion rates and the horizontal slices' vertical relationship, which is invalid in the incision case. This is because the incision erodes the deposits, making the time no longer fully determine the vertical facies distribution.

5.3 Summary of Chapter 5

This chapter presents the Fluvial GAN 3D simulator composed of the Fluvial GAN generator for 2D simulation and the SPADE-based conditional generator for 3D reconstruction, which is the product of this thesis. Due to the challenges of 3D facies modelling using GANs, I devised an alternative approach to realise the 3D reservoir facies modelling, reconstructing 3D volumes slice by slice. This 3D reconstruction program, FluvialGAN_3DR, requires a conditional GAN-based workflow using data pairs (an upper slice and a lower slice) to train a SPADE generator to predict an upper slice conditioned to a given lower slice. In this way, the Fluvial GAN 3D simulator can create facies models with an arbitrary number of slices/reservoir thickness in meters, one bottleneck of the previous 3D convolutional neural network-based GAN models. Also, the FluvialGAN_3DR training is less computationally costly than conventional 3D GAN training because (1) it deals with 2D data leading to a lighter GAN model regarding the number of learnable parameters, and (2) it disconnects the gradients, resulting in a lower demand of the GPU memory.

I developed two training enhancements, the multi-step and conditioning data decay enhancements, for FluvialGAN_3DR to improve its performance in the reconstruction process. The multi-step enhancement integrates the reconstruction into the GAN training workflow, which sets up four reconstruction steps with different target slices above the base slice. This enhancement trains GAN to avoid only focusing on predicting the target slice (upper slice) immediately above the given slice (lower slice). The conditioning data decay enhancement changes the conditioning data (lower slice) in the conditional GAN from the original slice directly below the current target to a composited slice containing all slices below the current target with a decay mechanism. So, the closer slices have higher weights, and the deeper slices have smaller weights in the conditioning data decay calculation. This enhancement improves the reconstructed samples' model diversity at their upper sections.

I heuristically coupled the avulsion rate group labels in GAN River-I with the avulsion rate factor (N_{av}) that reflects the vertical correlation between slices in a purely aggrading system and suggested that varying the N_{av} value provides an interpretable way of blending the modellers' knowledge into the reconstruction process. As N_{av} value is correlated to the avulsion rate, tuning N_{av} clearly means changing the avulsion rate setting based on the modellers' understanding. A smaller N_{av} value suggests a high avulsion rate setting, resulting in FluvialGAN_3DR producing more elongated channels. On the other hand, a bigger N_{av} value suggests a low avulsion rate setting, leading to FluvialGAN_3DR reconstructing more meandering channels. That relationship between N_{av} and geological patterns is consistent with the observed relationship between avulsion rate and geological patterns. Thus, the Fluvial GAN 3D simulator not only allows the modellers to choose the reservoir thickness but also enables tuning geological patterns based on their understanding of the modelled reservoir.

Chapter 6

GAN-based Approach Extension and Data Conditioning

In the earlier chapters, Fluvial GAN 3D proved the GAN-based approach could efficiently create 3D facies models of complex sedimentary systems. Previous chapters used meandering fluvial models to demonstrate GAN-based approaches' capability of learning geological patterns from process-based models, broadening GANs' application scope of subsurface modelling. However, Fluvial GAN 3D has a highly complex architecture and works on reservoirs with fixed lateral sizes.

Considering the facies model's different roles in subsurface modelling, Fluvial GAN 3D needs corresponding developments to fulfil real reservoir applications. For example, modellers often require the facies model to honour observed data. Thus, conditional simulation is more desirable, demanding data conditioning techniques to constrain Fluvial GAN 3D. If the facies model is involved in a model updating pipeline, the modeller perhaps prefers a small number of parameters in the optimisation. More gaps between Fluvial GAN 3D and real reservoir modelling need a thoughtful study and appropriate solutions.

This chapter preliminarily investigates how to extend Fluvial GAN to different reservoir applications. For simplicity, this chapter uses 2D examples to explore other techniques' effects on GAN-based approaches.

6.1 GAN-based Geological Parameterisation

Geological parameterisation is one important application of GAN-based approaches, which refers to parameterising large geological models with a small number of elements where the geological models can be sampled. Such that the modellers can tune their geomodels to dynamic data in a geologically consistent way. GAN's latent vector often obeys a pre-defined distribution, such as Gaussian distribution, making it suitable to work as a parameterisation method [Chan and Elsheikh, 2019]. This section uses the Fluvial GAN in section 4.2 as the baseline to study the effect of the latent vector on generation quality and size (please refer to the input vector in Figure 2.26 for an illustration of the latent vector of GANs).

6.1.1 Study of Latent Vector Size

This study decreases the latent vector size of Fluvial GAN to investigate the impact of a smaller latent size on Fluvial GAN's generation quality. A smaller number of parameters in the model updating pipeline is favourable because it reduces the time cost and makes the optimisation executable by many heuristic-based algorithms, such as genetic algorithm [Michalewicz and Schoenauer, 1996] and particle swarm optimization [Kennedy and Eberhart, 1995]. Table 6.1 summarise the four Fluvial GAN cases with different latent vector sizes used in this study. The rest settings remain the same as the Fluvial GAN's default configuration.

TABLE 6.1: Summary of Fluvial GANs' latent vector size

Case	Baseline	Case 1	Case 2	Case 3	Case 4
Latent Vector Size	128	64	32	16	8

After training, the trained generator randomly simulates 1000 realisations for each case for qualitative and quantitative evaluations. This study follows the same approach as in section 4.2.4 to compare different Fluvial GAN cases in connectivity, diversity and unrealistic feature occurrence. The above indicators demonstrate the impacts of the latent vector size on Fluvial GAN's generation quality regarding different considerations. Modellers, therefore, can balance the simulation quality and the number of latent parameters when they use GAN for geological model updating pipelines.

Based on the calculated sand proportion and connectivity, all four cases achieve comparable performance to the Fluvial GAN baseline. The data clouds of the four Fluvial GAN cases spread widely (see Figure 6.1). They largely cover the FLUMY data cloud, illustrating they capture the range of sand proportion and connectivity represented by the training dataset. Statistics show that decreasing latent vector size has a neglectable impact on sand proportion and limited (about 5% in mean connectivity) damage to sand connectivity (see Table 6.2).

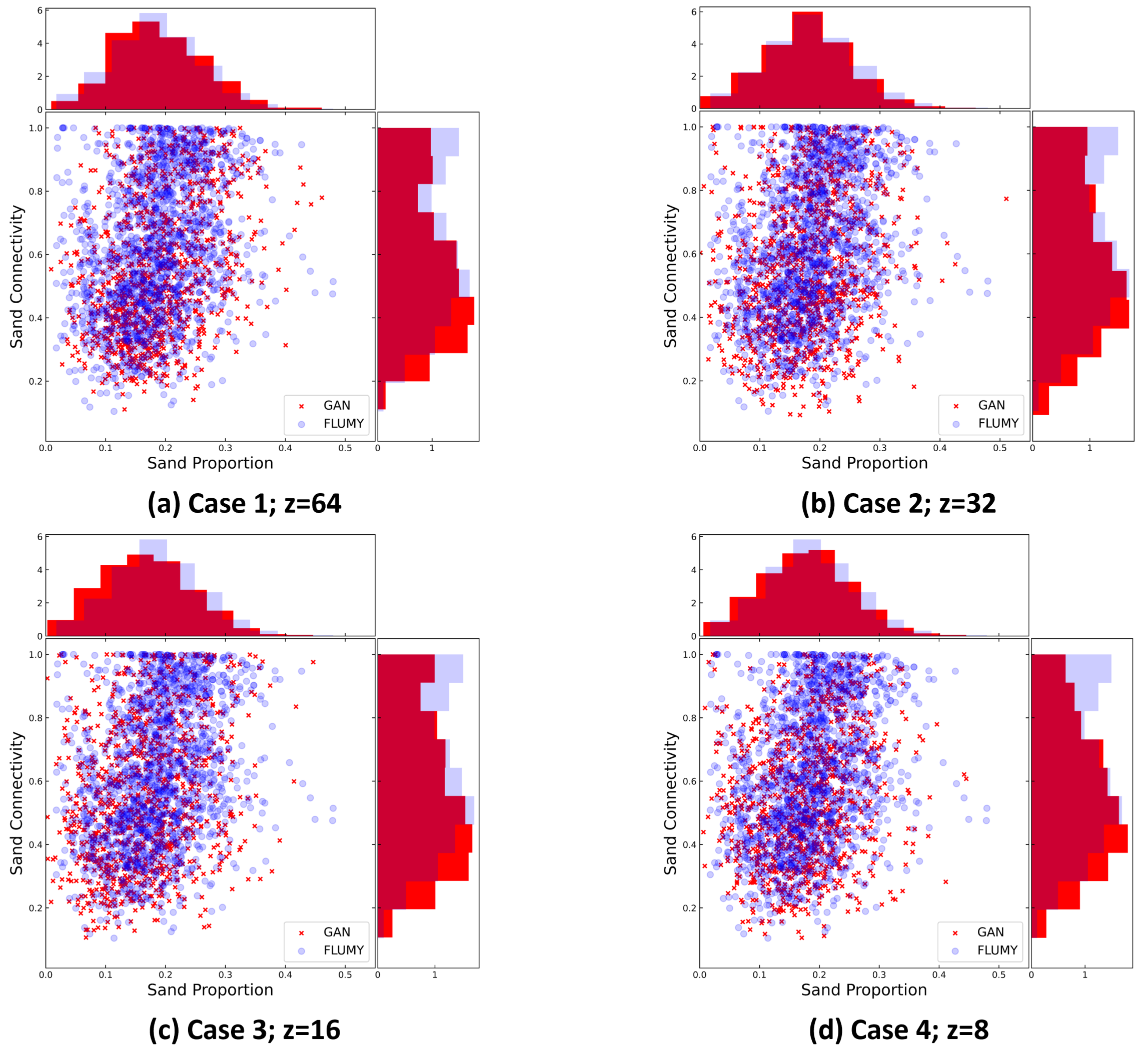


FIGURE 6.1: Plot of different Fluvial GAN sets' Sand Connectivity against Proportion. Here NTG in the x-axis denotes the proportion of sand. Blue points are FLUMY realisations. Red points are GAN realisations. (a)(b)(c)(d) are Case 1 to 4 in Table 6.1, respectively.

TABLE 6.2: Statistics of different Fluvial GAN sets' sand connectivity and proportion

Data Set	Latent Vector Size	Sand Proportion Mean	Sand Proportion Variance	Sand Connectivity Mean	Sand Connectivity Variance
FLUMY Subset	NA	0.19	0.005	0.62	0.049
Baseline	128	0.18	0.005	0.58	0.047
Case 1	64	0.19	0.005	0.56	0.049
Case 2	32	0.18	0.005	0.57	0.050
Case 3	16	0.17	0.006	0.56	0.051
Case 4	8	0.18	0.005	0.55	0.044

Then the plots of the sand connectivity probability along the lag distance, show a slight shift to lower connectivity probability when Fluvial GAN uses a smaller latent vector size (see Figure 6.2). All four Fluvial GANs present a wide range of sand connectivity probability, while the mean values and box plots indicate they might slightly underestimate the connectivity (see Figure 6.2). Though the Fluvial GAN baseline has this underestimation itself, the decreasing latent vector size seems to enlarge the difference. Nevertheless, the latent size has a limited impact on Fluvial GAN's performance regarding the sand connectivity probability.

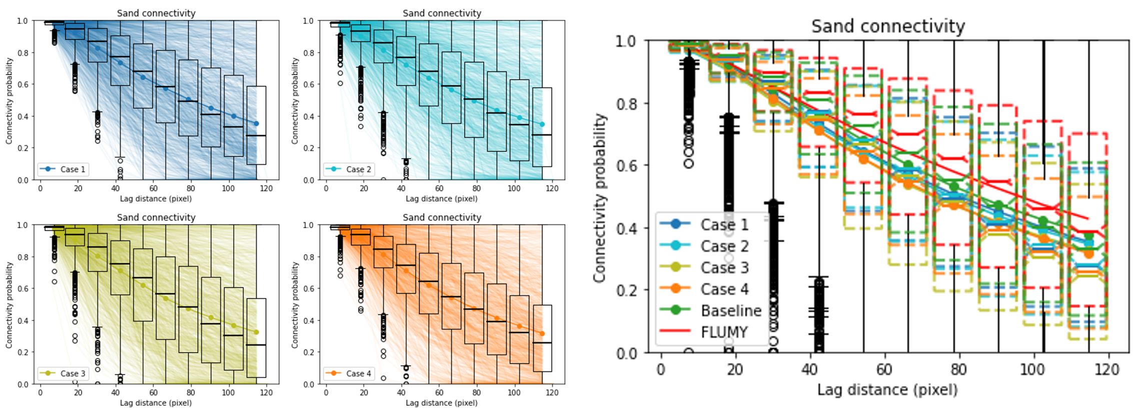


FIGURE 6.2: Sand connectivity probability curves. Blue curves are realisations from Case 1. Cyan curves are realisations from Case 2. Olive curves are realisations from Case 3. Orange curves are realisations from Case 4. Green curves are realisations from Fluvial GAN. Red curves are realisations from the FLUMY training dataset. The solid lines are mean values, and the black box and dash-lined boxes indicate 75%, medium, and 25% of the data sets.

UMAP visualisation illustrates Fluvial GANs having smaller latent sizes can still produce a diverse ensemble of GAN realisation comparable to GAN River-I ensemble diversity.

The red points of Fluvial GAN cases in the UMAP space exist in wide areas, largely overlapping the blue points (FLUMY realisations) cloud (see Figure 6.3). There is no apparent shrinkage of the data cloud spread in UMAP plots for cases 1 to 4.

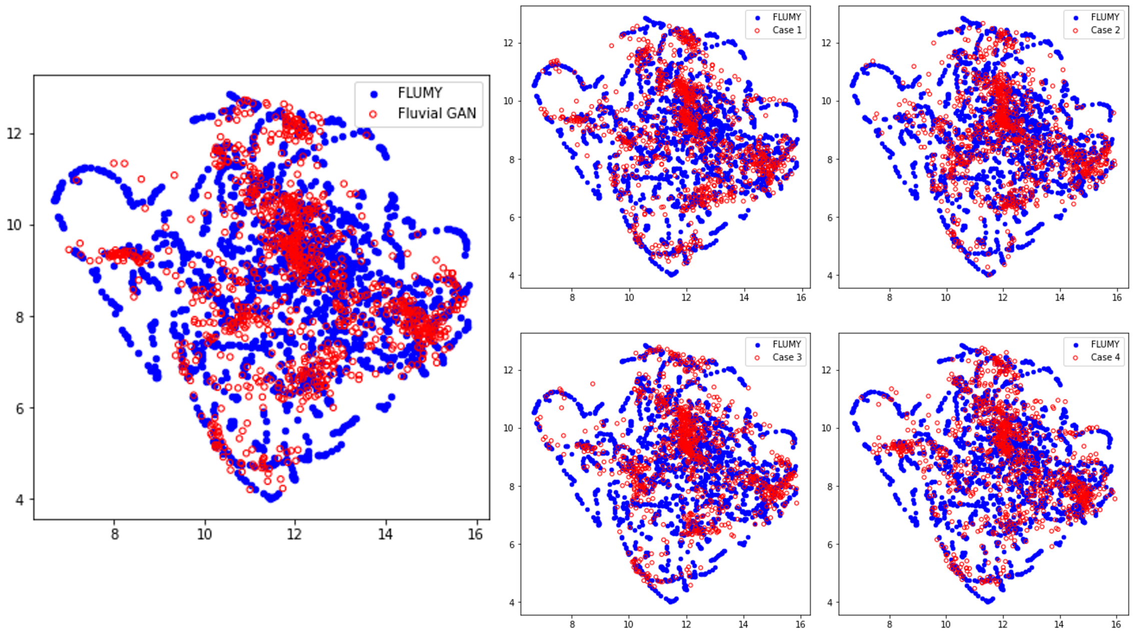


FIGURE 6.3: UMAP visualisation of FLUMY realisations against Fluvial GANs' realisations. Blue points are FLUMY realisations. Red points are Fluvial GANs' realisations.

Qualitative analysis reveals that the decreasing latent size impairs Fluvial GAN's performance in geological realism and artefact level. Based on visual interpretations of all realisations from the four Fluvial GAN cases, the occurrence of two major issues increases, the 'closed channel' pattern and the 'gridding effect' (see Table 6.3). More importantly, the 'gridding effect' level increases when the latent vector size is decreased, resulting in poor quality (see Figure 6.4).

TABLE 6.3: Occurrence-based indicators for different Fluvial GAN cases

Case	Latent Vector Size	'Closed channel' Occurrence	'Gridding effect' Occurrence
Baseline	128	0.7%	2.1%
Case 1	64	5.2%	1.9%
Case 2	32	1.3%	13.7%
Case 3	16	0.8%	7.1%
Case 4	8	1.9%	11.3%

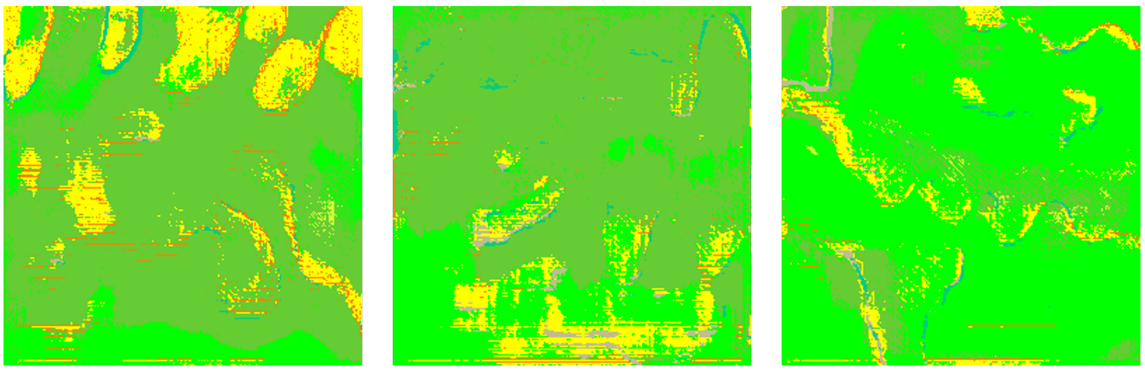


FIGURE 6.4: Poor quality facies models collected from Case 4.

6.1.2 Study of Enlarging Generation Size

Customising GANs' generation size remains challenging, impeding its quick implementation to different size reservoirs. Though Chapter 5 presented a conditional GAN-based approach to simulate facies models with flexible control of the thickness, resizing facies models laterally still relies on retraining some neural networks or changing the latent size of the generator. Particularly enlarging the generation size of a pre-train GAN is a helpful feature for the reuse of the GAN model in different-size reservoirs, as GAN generations are normally small in pixels.

This study tries to enlarge a pre-train Fluvial GAN's generation size proportionally by increasing the latent size. Earlier research proved that deep convolution-based GAN containing no fully connected layer could produce different size data [Laloy et al., 2018, Song et al., 2022]. Still, this method can only vary generation size proportionally while not customising the width and height of the generated data.

This study tests a fully convolution-based variant of Fluvial GAN by replacing the dense layer with transposed convolutional layers. The new generator architecture allows a varied input size and proportionally projects the input vector to data space (see Table 6.4 for the generator architecture details). The rest configuration remains the same as the standard Fluvial GAN.

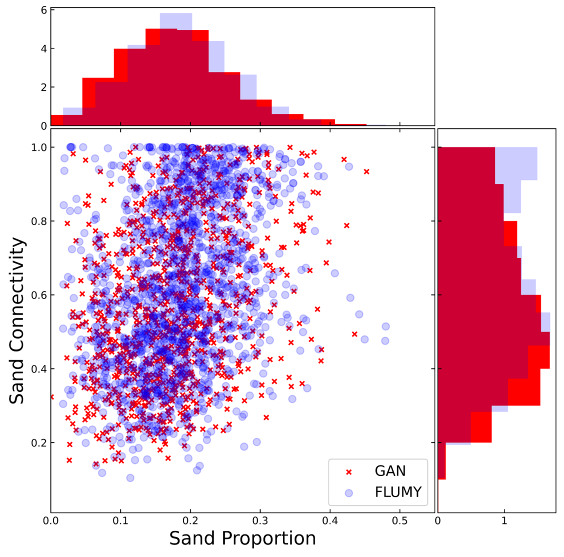
To understand the impact of the change in the generator architecture, this study takes the same analyses as before, calculating connectivity to evaluate quantitatively, using UMAP to visualise the generation diversity, and interpreting unrealistic features visually. GAN

Table 6.4: The Architecture of Fluvial GAN Generator without Dense Layer

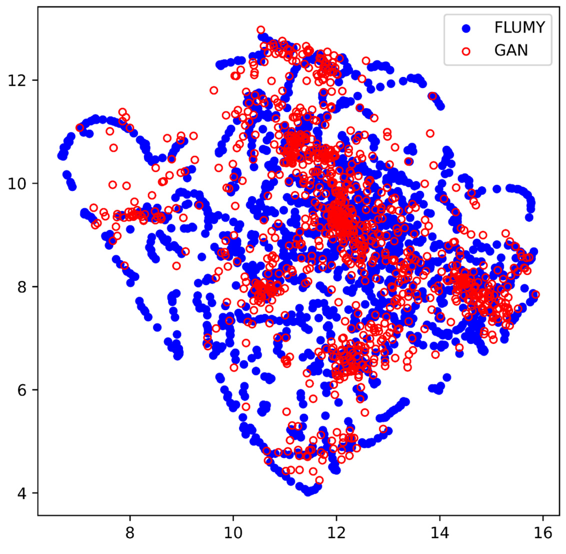
Layer	State size
Latent Vector	(Batch size, 128, 1, 1)
TransposeConvNN Block (stride=1,padding=0)	(Batch size, 512, 4, 4)
TransposeConvNN Block (stride=2,padding=1)	(Batch size, 512, 8, 8)
NN Block, Up sample (2)	(Batch size, 512, 16, 16)
NN Block, Up sample (2)	(Batch size, 256, 32, 32)
NN Block, Up sample (2)	(Batch size, 128, 64, 64)
NN Block, Up sample (2)	(Batch size, 64, 128, 128)
NN Block, Up sample (2)	(Batch size, 32, 256, 256)
NN Block, 3X3Conv-7, SoftMax	(Batch size, 7, 256, 256)

3X3Conv-7 denotes 3X3 convolutional layer with seven filters. NN Block refers to the neural networks sequentially composed of a batch normalisation, a leaky ReLU, and a 3X3 convolutional layer. TransposeConvNN Block refers to the neural networks sequentially composed of a 4X4 transposed convolutional layer, batch normalisation and a ReLU.

can still produce diverse facies models according to the connectivity plot and UMAP visualisation (see Figure 6.5). GAN generations show a good coverage of sand proportion and connectivity ranges represented by GAN River-I. The data spread in the UMAP space also indicates the generation diversity of the fully convolution-based variant is comparable to the standard Fluvial GAN (see Figure 6.3 and 6.5 b).



(a) The sand connectivity VS NTG plot



(b) Visualisation in the UMAP space

Figure 6.5: The sand connectivity against proportion plot and UMAP visualisation of FLUMY realisations against Fluvial GANs' realisations. Blue points are FLUMY realisations. Red points are Fluvial GANs' realisations.

On the other hand, the occurrence and the level of unrealistic features increase to some degree based on visual interpretation compared to the standard Fluvial GAN. Same to Case 4 in section 6.1.1, the fully convolution-based variant creates poor quality facies models containing severe 'gridding effect' (see Figure 6.6), which are unseen in the standard Fluvial GAN. Based on the occurrence, the 'closed channel' probability increases to 4.4%, and the 'gridding effect' probability increases to 3.5%, but both are within an acceptable value (smaller than 5%).

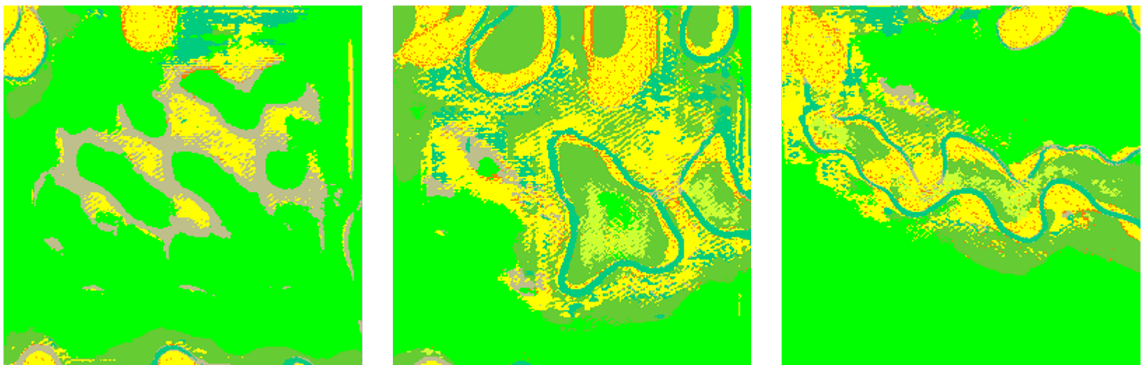


FIGURE 6.6: Poor quality facies models collected from the fully convolution-based variant.

These analyses demonstrate the fully convolution-based variant still performs well though more likely to produce unrealistic features than the standard Fluvial GAN. Most generations show plausible diverse meandering patterns, indicating this Fluvial GAN variant learns from the training dataset and is free of mode collapse. This result supports further variation in generation sizes by modulating the latent vector of this pre-trained generator, which is the main objective of this study.

The fully convolution-based variant allows a varied spatial size (height and width) of the latent vector to produce different size generations. The convolutional neural network block in Fluvial GAN doesn't change the spatial size, while only the upsampling layer doubles the size in each spatial dimension. So, the GAN generation size is proportional (32 times) to the second transposed convolutional layer's output size. According to Equation 2.17, the spatial size relationship between the latent vector and GAN generation size is given by Table 6.5.

The first experiment feeds random latent vectors with a larger spatial size into the pre-trained fully convolution-based generator to create bigger facies models. When the latent

Table 6.5: Occurrence-based indicators for different Fluvial GAN cases

Latent Spatial Size	First Transposed Convolution Output Size	Second Transposed Convolution Output Size	GAN Output Spatial Size
1×1	4×4	8×8	256×256
2×2	5×5	10×10	320×320
3×3	6×6	12×12	384×384
4×4	7×7	14×14	448×448
5×5	8×8	16×16	512×512

vector spatial size is 2, the pre-trained generator can produce both plausible and implausible facies models (see Figure 6.7). Though those models cover a wide range of connectivity, their sand proportions are generally bigger than the training dataset because GAN tends to produce big blocks of point bars (yellow facies), such as the pattern in Figure 6.7 d. When the latent vector spatial size increases to 5, the generation quality worsens, and the sand proportion becomes bigger (see Figure 6.8). Most generations contain broken artefacts or big chunks of implausible point bars, though many facies models still have some plausible meandering patterns as patches in the modelled domain.

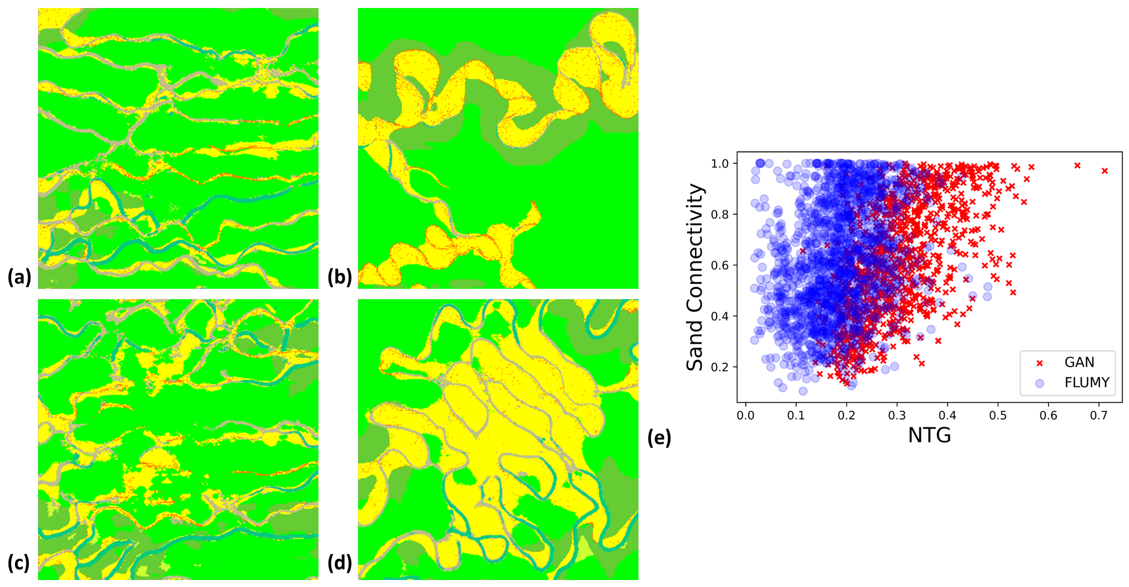


Figure 6.7: Results of the fully convolution-based variant producing larger facies models by feeding random latent vector whose spatial size is 2. (a) A plausible 320×320 less developed meandering model. (b) A plausible 320×320 lateral developed meandering model. (c) An implausible 320×320 less developed meandering model with poor quality and broken geo-bodies. (d) An implausible 320×320 lateral developed meandering model. (e) The sand connectivity against proportion plot of 1000 320×320 random realisations.

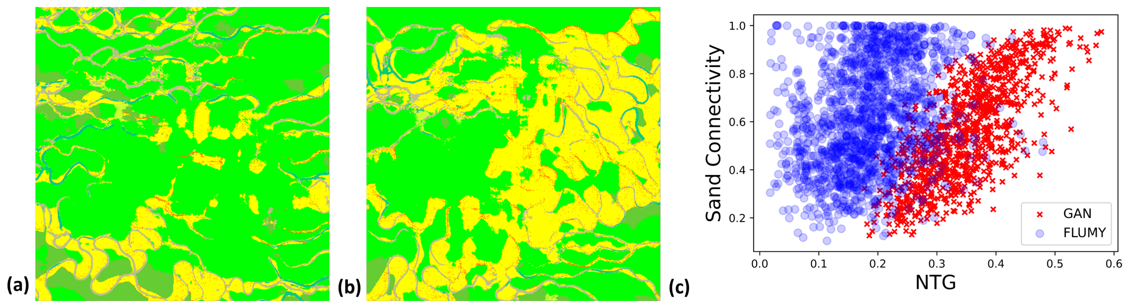


FIGURE 6.8: Results of the fully convolution-based variant producing larger facies models by feeding random latent vector whose spatial size is 5. (a) An implausible 512×512 less developed meandering model. (b) An implausible 512×512 lateral developed meandering model. (c) The sand connectivity against proportion plot of 1000 512×512 random realisations.

The second experiment expands the latent vector by repeating its values in spatial dimensions to enlarge the facies models to a bigger size. In this way, the actual number of GAN's latent parameters remains unchanged as the standard when it needs to generate bigger-size facies models. The generations gradually show more undesired patterns and have a bigger range of sand proportion with the increase of the latent vector spatial size (see Figure 6.9). The enlarged facies models present a clear trace of repeated patterns (Figure 6.9 b and c) corresponding to their standard size facies models (Figure 6.9 a). Similar to the results of feeding larger random latent vectors, the enlarged facies models are mostly implausible with artefacts and stacked point bars. Only a few examples show positive results that the enlarged facies models contain plausible meandering patterns, though they have strong duplicated patterns (see Figure 6.10).

This study demonstrates enlarging the Fluvial GAN generation size by changing its latent vector size performs undesired and needs further improvements to tackle this challenge. Using fully convolution-based GAN to produce different-size outputs is theoretically practical but difficult to handle the process-based meandering patterns. Fluvial GAN tends to make implausible facies models When feeding a bigger-size latent vector than it is trained. How to stabilise Fluvial GAN's performance when producing larger-size facies models based on a pre-trained Fluvial GAN leaves an unsolved problem for future investigation.

As for creating large-size data, Karras et al. [2017] developed a GAN training method for super-resolution image generation, called progressive growing, which has been applied to reservoir facies modelling [Song et al., 2021a,b, 2022]. This method can help GAN create

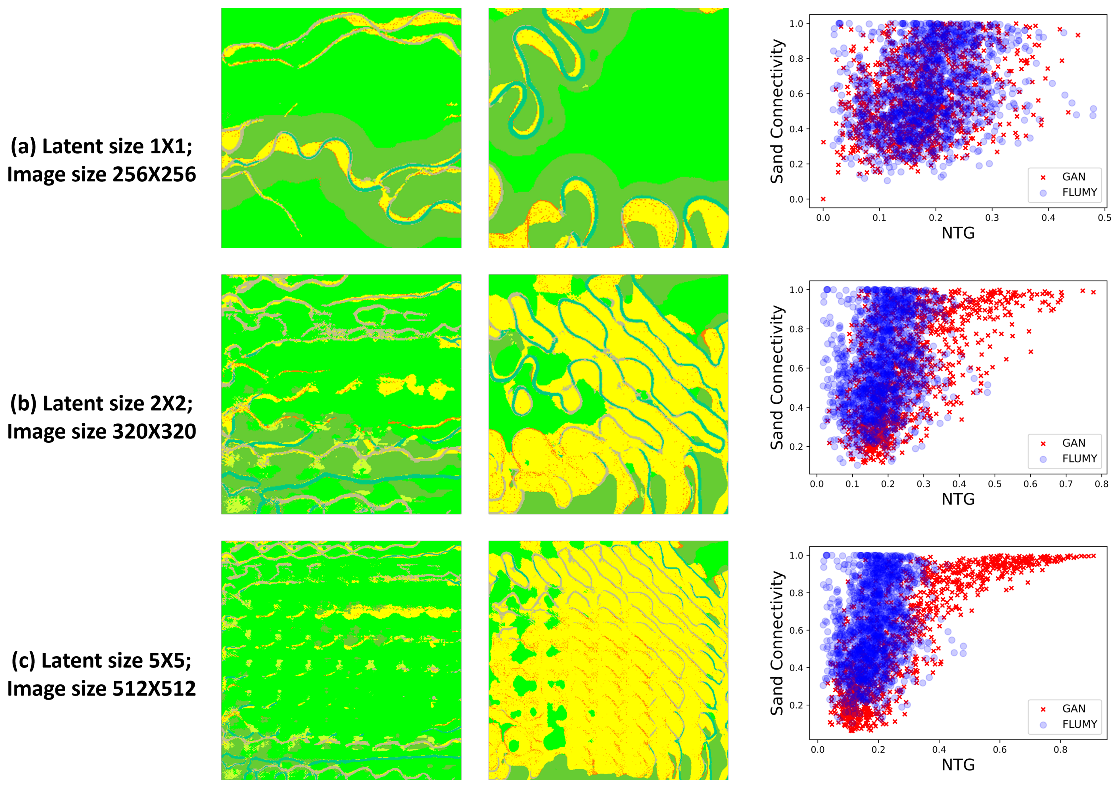


FIGURE 6.9: Results of the fully convolution-based variant producing larger facies models by expanding the latent vector. (a) Meandering models and the sand connectivity against proportion plot from 1000 1×1 latent vectors. (b) Meandering models and the sand connectivity against proportion plot from 1000 2×2 latent vectors with repeated values of the 1×1 latent vectors. (c) Meandering models and the sand connectivity against proportion plot from 1000 5×5 latent vectors with repeated values of the 1×1 latent vectors.

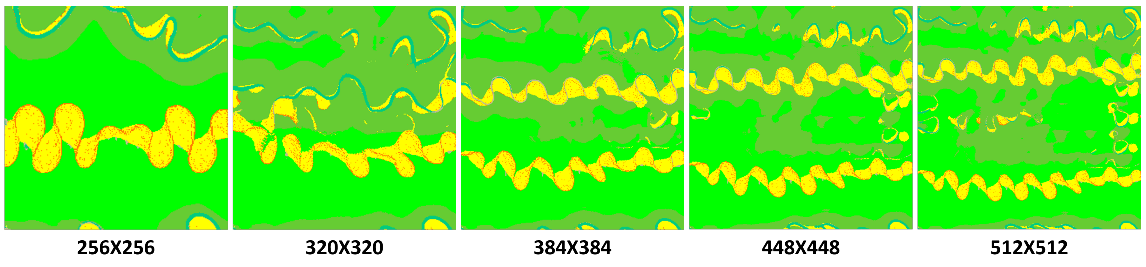


FIGURE 6.10: An example of the fully convolution-based variant successfully enlarging facies models by repeating a latent vector to expand it from 1×1 to 5×5.

big (in terms of pixels) images by using the same images from low to high resolution to train the GAN model hierarchically. This super-resolution task shares the same challenge as modelling large-size reservoirs – generating large-size (in terms of pixels) data. The main difference is that the super-resolution task aims to produce a higher-resolution image of the same image currently in a lower resolution. In contrast, reservoir modellers want to produce a larger size reservoir model that can share similar or partially the same patterns but a different (in terms of the whole modelled area) reservoir. Based on [Song et al., 2022], their fully convolutional neural network-based GAN model trained with progressive growing successfully produces a larger binary 3D cave model than the training data by enlarging the latent size. Therefore, the progressive growing method can potentially stabilise Fluvial GAN with a fully convolutional neural network design.

6.2 Data Conditioning

Conditioning GAN to observed data is highly required in real-field reservoir modelling but is particularly challenging due to GAN's competitive learning mechanism and forward sequential model structure. Earlier GAN applications to conditional simulation used post-processing approaches, e.g. inference networks, and different conditional GANs to condition the simulations to wells data, probability maps and global features [Chan and Elsheikh, 2019, Song et al., 2021a, Zhang et al., 2019b]. Please refer to Section 2.3.3 for the overview of those conditioning techniques' pros and cons.

Post-processing allows a pre-trained unconditional GAN to simulate conditional realisations. Previous research developed different conditioning frameworks using the latent vector of a pre-trained GAN generator. This method separates the conditional simulation into two parts: geological patterns learning and getting the realisations to honour the observed data. One advantage of this method is that it allows the pre-trained GAN to be reusable in different reservoirs with the same size and sedimentary setup, which saves time in creating the training dataset and training the GAN model.

Conditional GAN-based frameworks can soft condition GAN generations to multiple types of data but have a general problem: they couple geological pattern learning and data conditioning. Most conditional GAN training processes train GANs to infer target data based on the given conditioning data. This results in GAN relying on conditioning data with strict type and range, which means GAN doesn't learn adequate geological features from the training data but learns to infer geological models from the features provided by the conditioning data. One drawback of this approach is that the user has to re-train the whole conditional GAN or find a way to convert all other conditioning data into the same format used in the current conditional GAN when the type of conditioning data available changes.

This section includes two studies on conditioning Fluvial GAN to nine points observation using the methods presented in Chan and Elsheikh [2019] and Zhang et al. [2019b]. Appendix A presents a preliminary work about a conditional GAN-based framework that separates the data conditioning from the traditional conditional GAN learning process while this approach is still immature and problematic.

Chan and Elsheikh [2019] designed a GAN-based conditional simulator using a neural network-based inference network to project a new latent vector to a pre-trained GAN latent vector (see Figure 6.11). The training of the inference network aims to minimize the KL divergence from the conditional latent vector (the Bayesian posterior) distribution to the distribution density of this output vector of the inference network. Chan and Elsheikh [2019] deduced the KL divergence calculation of the inference network, and the final form is given by Equation 6.1 (please refer to Chan and Elsheikh [2019] for more details of the deduction)

$$D_{KL}(q_\phi || p(z|d)) = \mathbb{E}_{z \sim q_\phi} L(z) - H(q_\phi) \tag{6.1}$$

where the first element, $\mathbb{E}_{z \sim q_\phi} L(z)$, calculates the mean loss (log posterior of the conditional latent vector), $-\log p(z|d)$, by adding the difference between observed data and sampled realisation at the well locations, $||G(z)_{obs} - d||^2$, and the prior distribution density (Gaussian distribution). The second element, $H(q_\phi)$, is a k-nearest-neighbor-based estimation of the negative entropy $-\mathbb{E}_{z \sim q_\phi} \log q_\phi(z)$.

The first study trains the same inference network with the default settings to condition the pre-trained Fluvial GAN generator to the nine points observation (see Figure 6.15 where the orange dots denote sand and the blue dots denote shale). Compared to the implementation in Chan and Elsheikh [2019], this program sums the sand-prone facies relative probabilities (Fluvial GAN generator's output) instead of converting the observed

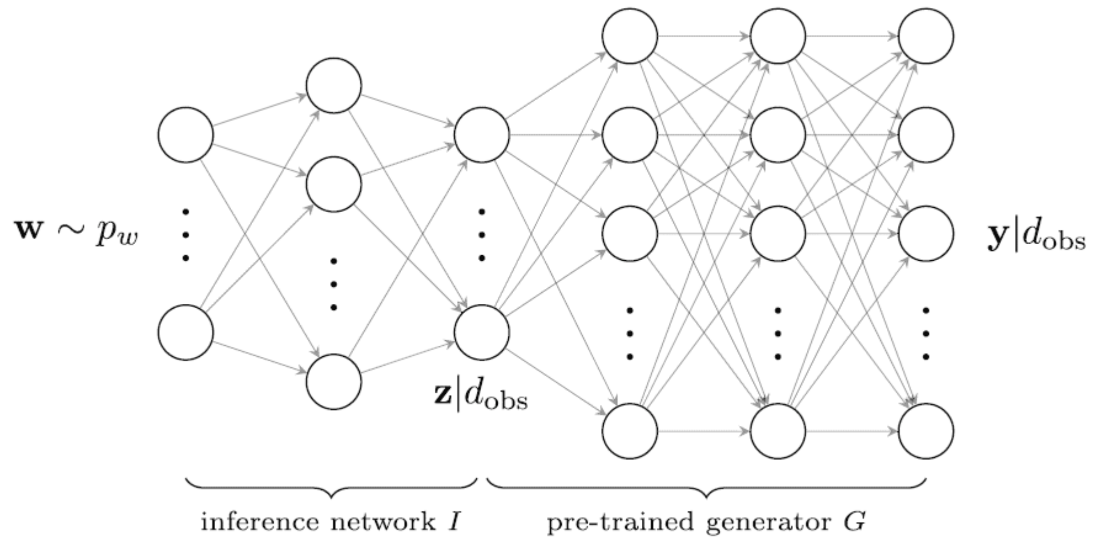

$\mathbf{w} \sim p_w$
$\mathbf{z}|d_{\text{obs}}$
$\mathbf{y}|d_{\text{obs}}$
inference network I
pre-trained generator G

FIGURE 6.11: A schematic diagram of the inference network method presented in Chan and Elsheikh [2019].

data (sand and shale) to -1 and 1 during the data mismatch calculation, $||G(z)_{obs} - d||^2$. Also, due to the graphic card limitation, the batch size reduces from 32 to 16.

Within 200,000 iterations, however, the framework fails to converge, and the conditional Fluvial GAN's generations, therefore, do not perfectly honour the observed data, though the accuracy of data matching at well locations has a clear increase. All three loss values in Equation 6.1 show a limited change during training (see Figure 6.12 a). The mean (negative) log posterior of the latent vector (output of the inference network), whose prior is a Gaussian distribution, decreases due to the decrease of the prior distribution density and the mismatch between the observed data and conditional realisations, $||G(z)_{obs} - d||^2$ (see Figure 6.12 b).

A further study investigates the effect of this conditioning framework by comparing the accuracy and model diversity between the unconditional Fluvial GAN and the inference network-based conditional Fluvial GAN. Fluvial GAN randomly simulates 1000 realisations by sampling 1000 latent vectors as the control group. The same latent vectors feed into the inference network-based conditional Fluvial GAN to create 1000 realisations as the experimental group. Compared to the unconditional realisation set, the mean accuracy of the conditional realisation set's data match at the well locations increases from 60% to 72%. The conditioned realisation set still shows good coverage of the sand connectivity

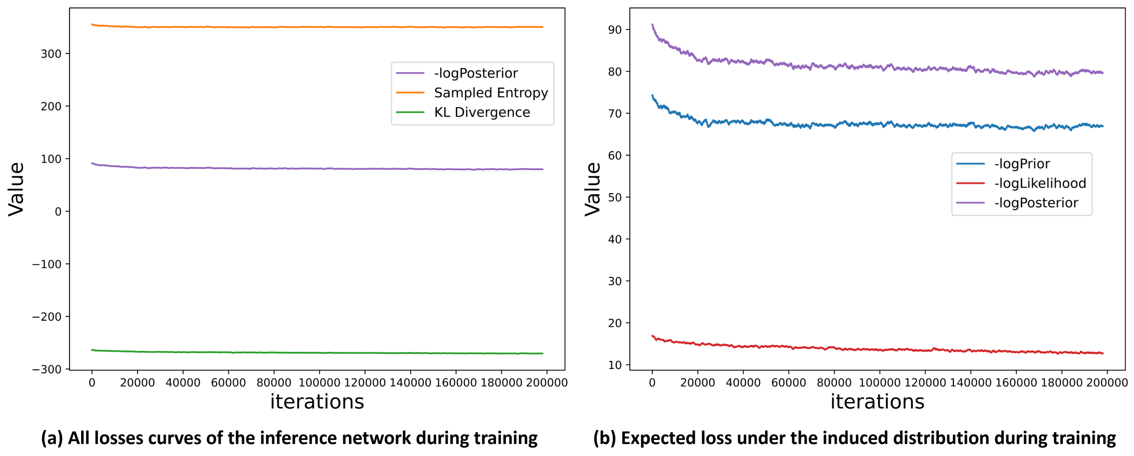


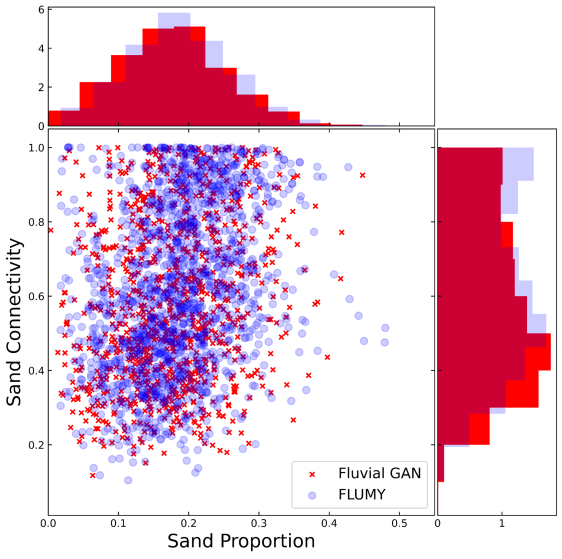
FIGURE 6.12: Loss curves during the training of the inference network. (a) Curves of the three elements in Equation 6.1. The purple, orange and green curves show changes of $\mathbb{E}_{z\sim q_\phi}L(z)$, $H(q_\phi)$, and $D_{KL}(q_\phi||p(z|d))$, respectively. (b) A zoom-in view of the expected loss (purple curve), $\mathbb{E}_{z\sim q_\phi}L(z)$, and its two elements,$-\log Prior$ (blue curve) and $-\log Likelihood$ (red curve).

and proportion ranges (see Figure 6.13). According to the UMAP visualization, the conditional realisation set has a good model diversity (see Figure 6.14). A visual check of the conditional realisation set indicates that the conditional generator can produce diverse plausible meandering facies models, though it does not perfectly match the observed data at well locations (see Figure 6.15). Those plots prove this conditioning framework doesn't undermine the realism and diversity learned by Fluvial GAN.

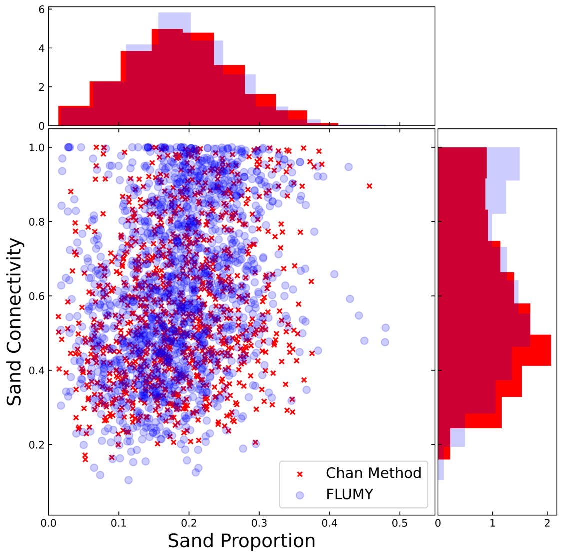
Zhang et al. [2019b] developed an optimisation-based conditioning framework that searches latent vectors to minimise the mean distance between the mismatched observed data and its nearest corresponding facies' location. The object function of the optimisation-based conditioning framework consists of two elements: perceptual loss and contextual loss. The perceptual loss is a neural network-based score, which is the binary cross-entropy loss for the generator, $\log(1 - D(G(z)))$. The contextual loss is the sum of the shortest distance between well locations with mismatched data and the locations of the correct facies, given by Equation 6.2 [Zhang et al., 2019b]:

$$L_c(Z|I_1 = i_1, I_2 = i_2, ..., I_M = i_m) = \sum_{K=1}^{K}\sum_{d=1}^{M} \min ||y(i^{(k)}(G(z))) - y(i_d^{(k)})||_1 \qquad (6.2)$$

where K is the facies number, M is the number of well data, I is the generated realisation, i is facies at the well location d, y refers to the locations of all pixels with the corresponding

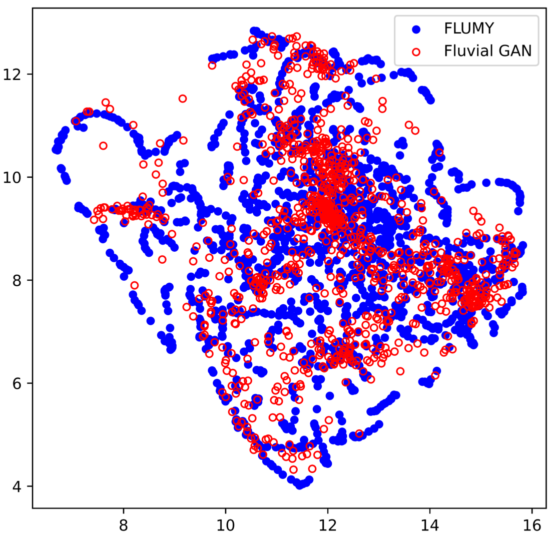


(a) Sand connectivity VS proportion plot of 1000 unconditional Fluvial GAN realisations

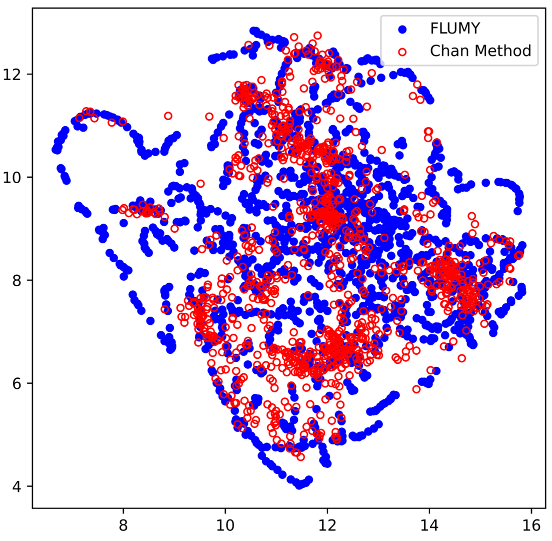


(b) Sand connectivity VS proportion plot of 1000 Fluvial GAN realisations conditioned to well data

FIGURE 6.13: Comparison of the sand connectivity against proportion plots between unconditional realisations from Fluvial GAN and conditional realisations from the inference network-based conditional Fluvial GAN generator.



(a) UMAP visualisation of 1000 unconditional Fluvial GAN realisations



(b) UMAP visualisation of 1000 Fluvial GAN realisations conditioned to well data

FIGURE 6.14: Comparison of the UMAP visualisations between unconditional realisations from Fluvial GAN and conditional realisations from the inference network-based conditional Fluvial GAN generator.

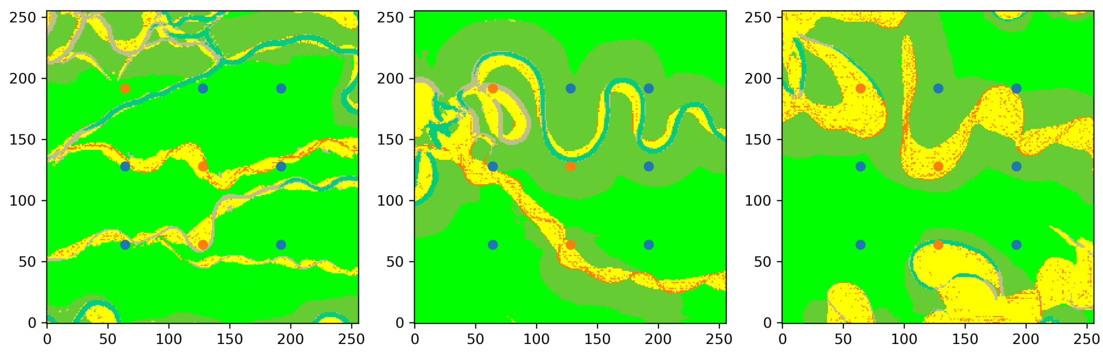
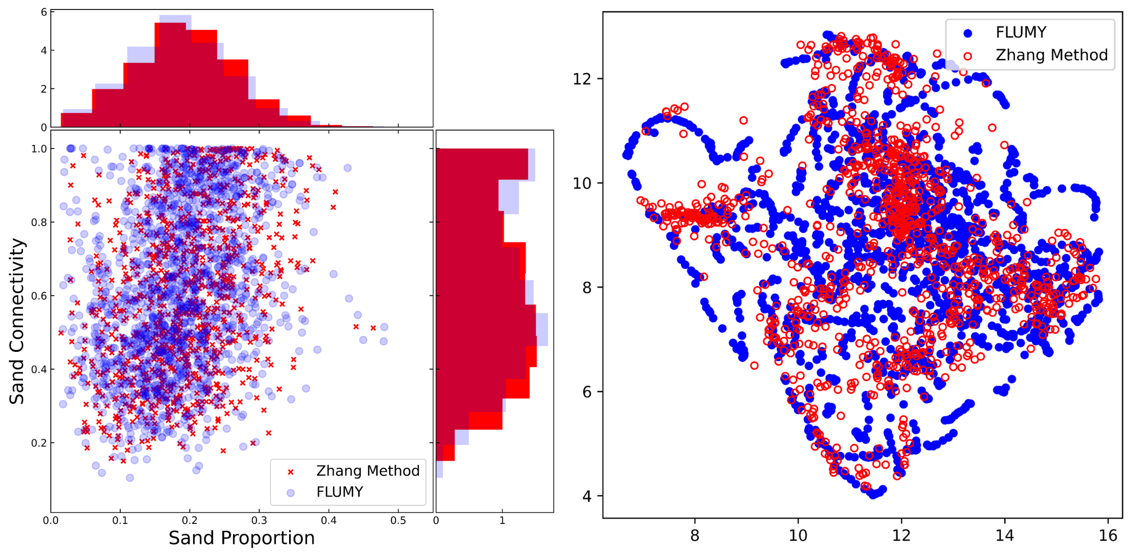


FIGURE 6.15: Conditional realisations with good data match observed data at well locations. The orange dots denote sand, and the blue dots denote shale.

facies, G is the pre-trained generator, and Z is the latent vector.

To integrate Fluvial GAN into the optimisation-based conditioning framework, a couple of changes to the original implementation replace the DCGAN architecture with the Fluvial GAN. First, the perceptual loss calculation needs to process through the PatchGAN discriminator and uses the mean values of the output collection of the discriminator instead of a single value as its input. As Fluvial GAN's loss function is hinge loss, the perceptual loss, therefore, needs to calculate $-D(G(z))$ instead of $\log(1 - D(G(z)$. Secondly, the output of the Fluvial GAN generator is the relative probability maps for each facies instead of a single image with continuous values. So, the conversion from the Fluvial GAN generator output to the binary realisation is approximated by summing all relative probabilities of sand-prone facies. Thirdly, this study tests the framework using two implementations: one uses both perceptual and contextual loss to optimise the latent vector, and the other only uses contextual loss to optimise the latent vector. The reason for having the second implementation is that the original implementation in Zhang et al. [2019b] gave a very big weight to the contextual loss ($\lambda = 1000$), which indicates a relatively low impact of the perceptual loss on the optimisation. So, the second implementation excludes the perceptual loss term that relies on a significantly different discriminator from its original implementation. The same 1000 latent vectors used in testing the inference network-based conditioning framework work as the initial latent vectors of this optimisation-based conditioning framework. Following Zhang et al. [2019b], each optimisation iterates 1500 times, timing about 10 min for a single GPU RTX3090 and a single 3.6 GHz CPU Intel(R) Xeon(R) W-2133.

Though the conditional realisations still present plausible patterns, both implementations of this framework show a limited accuracy increase in data match. The mean accuracy of the data matches at the well locations increases from 60% to 62% after optimising the latent vectors. Based on the visual and quantitative assessment, this framework also well preserves the geological realism and model diversity learned by Fluvial GAN. The ensemble of conditional realisations covers a wide range of NTG and connectivity (see Figure 6.16 a) The UMAP visualisation also supports that this conditioning framework does not sacrifice the generation diversity (see Figure 6.16 b).



(a) Sand connectivity VS proportion plot of 1000 Fluvial GAN realisations conditioned to well data

(b) UMAP visualisation of 1000 Fluvial GAN realisations conditioned to well data

FIGURE 6.16: The sand connectivity against proportion plot and UMAP visualisation of conditional realisations from the optimisation-based conditioning framework.

Two reasons account for the low accuracy increase in this Fluvial GAN conditional realisation set produced by the optimisation-based conditioning framework.

First, stochasticity strongly dominates this framework in the latent vector initialisation and optimisation. Here is an example that both unconditional realisations with randomly initialised latent vectors match five points in nine locations (see the first column of Figure 6.17). After running 1500 iterations of the optimisation, the conditional realisation produced by the latent vector with a bad initialisation doesn't have a better data match, while the one with a good initialisation improves the number of points matched to the conditioning data from five to seven (see the second column of Figure 6.17). This comparison illustrates that a good initial guess strongly impacts the optimisation result, which is one potential shortcoming of gradient-based optimisers that they are local search algorithms whose performance is affected by the 'starting point' [Chapelle and Wu, 2010, Kim et al., 2007].

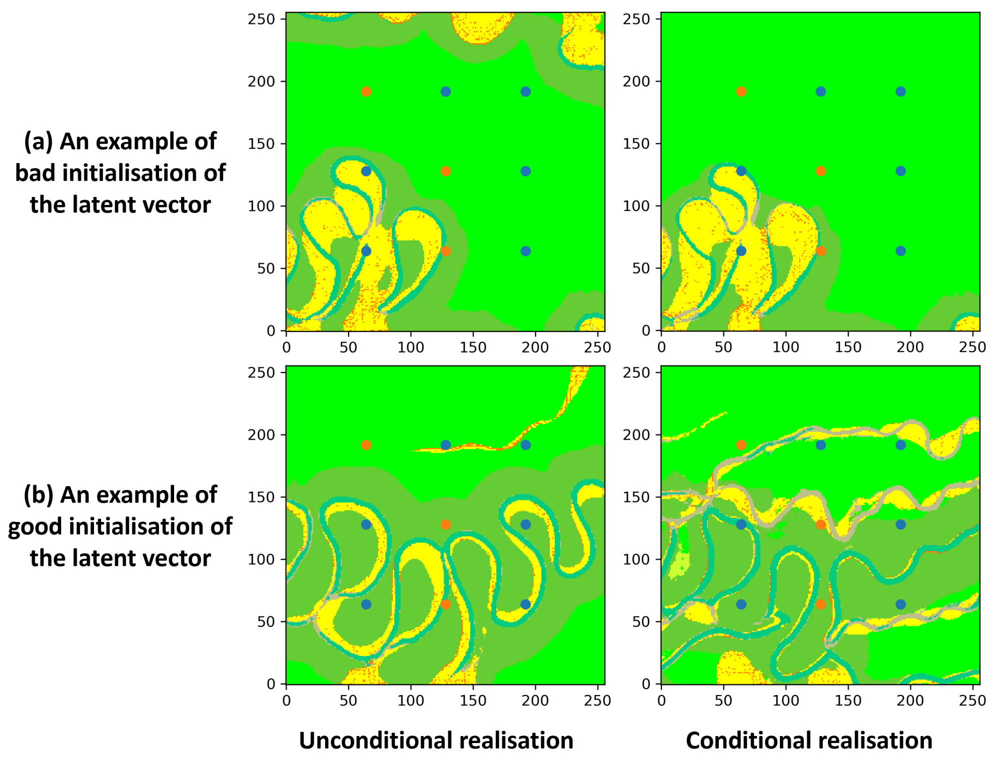


FIGURE 6.17: Illustration of the stochastic effect on the optimisation-based conditioning framework. The first column is the unconditional realisation from Fluvial GAN using a randomly initiated latent vector. The second column is the conditional realisation from Fluvial GAN using the optimised latent vector from the optimisation-based conditioning framework using the same latent vector in the first column as the input. (a) an example of a bad initialisation of the latent vector. (b) an example of a good initialisation of the latent vector.

Secondly, Fluvial GAN is more complex than previous GAN applications, and the design of merging Fluvial GAN into the optimisation-based conditioning framework lacks proper adjustment. Fluvial GAN has different model architecture, which may require further tuning the configurations of the optimisation-based conditioning framework by broad tests on hyper-parameters of this framework. Also, the method in Zhang et al. [2019b] is a gradient-based multivariate optimisation, using an Adam to minimise two objectives.

Many techniques, e.g. regularisation and gradient penalty, which may improve the optimisation performance, e.g. smoothing spike loss, are not applied to this optimisation-based conditioning framework. In addition, the original code of the official implementation in Zhang et al. [2019b] is not available online. Therefore, this implementation of the optimisation-based conditioning framework is limited to the author's understanding of the paper Zhang et al. [2019b].

6.3 Summary of Chapter 6

This chapter presented several extensions to further develop the Fluvial GAN 3D simulator, catering to the demands for applying it to different reservoir modelling applications. For simplicity, I used Fluvial GAN for 2D facies modelling to investigate other GAN configuration choices, e.g. varying the latent vector, and how to merge other GAN techniques, e.g. data conditioning, with Fluvial GAN. For use as a geological parameterisation, a smaller latent vector size of Fluvial GAN benefits the optimisations in the model updating pipeline. However, a smaller latent vector size worsens the Fluvial GAN's performance in geological realism based on visual interpretation. Thus, further efforts on optimising the Fluvial GAN's architecture are necessary if the modellers require a smaller latent vector size. Defining the facies models' size post-training is still challenging for most GAN applications. Though the Fluvial GAN 3D simulator allows defining the reservoir thickness due to its reconstruction nature, the Fluvial GAN (2D) is inflexible to change its lateral sizes. Based on the preliminary experiments in this chapter, excluding the fully connected layer does enable Fluvial GAN to vary its generation size proportionally. However, this method greatly increases the latent vector size and produces undesirable realisations when using it to enlarge the generation size. Thus, more research on this problem is essential to bring GANs to real reservoir applications. For data conditioning, both inference network-based and optimisation-based conditioning framework work as post-processing and don't undermine the model diversity of Fluvial GAN. However, these two methods either have unimpressive or no apparent improvements in point data matching accuracy. Therefore, how to condition Fluvial GAN to well data remains a future study. Overall, the Fluvial GAN model shows good compatibility with different GAN techniques proposed by other researchers, though all need further developments to improve the performance.

Chapter 7

Conclusions and Perspectives

7.1 Conclusions

This thesis demonstrated that generative adversarial networks (GANs) could learn meandering fluvial patterns produced by a complex process-based model. The process-based model, FLUMY, creates plausible facies models by simulating geological processes, such as channel migration, avulsion, levee breach, deposit sedimentation, etc. Those facies models contain multiple facies with complex and diverse geometries, reflecting the plausible complexity of subsurface reservoirs. Whether deep generative models, such as GANs, can learn to produce geological patterns at the process-based model level of complexity deserves a thoughtful investigation. This extension of GANs in facies modelling would help incorporate geological knowledge, concepts and interpretations into geo-modelling.

This PhD project developed a bespoke GAN-based framework for 3D facies modelling, Fluvial GAN 3D, to tackle identified challenges in GAN learning meandering fluvial geology. Based on a series of studies, several algorithms and techniques successfully improved the GAN learning performance undermined by challenges related to multi-facies, meandering patterns and data size. A new GAN variant, Fluvial GAN, can accurately reproduce 2D geometrical features of meandering fluvial facies and capture facies transition along the channel centreline. Based on the Fluvial GAN, a reconstruction-based conditional GAN framework, FluvialGAN_3DR, successfully learns to create 3D models layer by layer from bottom to top, conditioning each new layer on the one below. This feature allows the users to set up the reservoir thickness and guarantees geological consistency vertically. The pre-trained Fluvial GAN generator and the pre-trained SPADE generator-based reconstruction process in FluvialGAN_3DR compose the Fluvial GAN 3D simulator that can create 3D facies models with the user-defined reservoir thickness in meters/slices.

The studies and results in the preceding chapters support the novel findings of the thesis below:

1. GAN River-I dataset provides a pre-canned sterner benchmark to deep generative models' applications to facies modelling at the process-based model level. The facies models in GAN River-I are from a process-based model for meandering fluvial reservoirs, FLUMY, which can simulate diverse types of meandering fluvial geology by tuning the parameters related to geological processes. By selecting different configurations in the avulsion process while controlling the net-to-gross (NTG), GAN River-I contains various low NTG scenarios, including ribbon-type and sheet-type sand-bodies. The low NTG reduces the amalgamation and makes it easier to identify individual channels to evaluate GANs visually. Also, due to the low NTG setting, GAN River-I covers a wide range of sand-body connectivity, which is one of the crucial properties in reservoir modelling. Therefore, the connectivity and uncertainty in GAN River-I can be one quantitative indicator of GAN generations.

2. PatchGAN discriminator is a recommended model architecture of GANs when the training data has a large spatial size. High resolution and large reservoir size may require the spatial size of the facies model to be bigger than the spatial size of the training data used in common deep generative models (64×64 or 128×128 pixels). Though PatchGAN was originally a model structure in a conditional GAN for image-to-image translation, PatchGAN can also work as a generative model by replacing the conditional generator with an ordinary generator, for example, a deep convolutional generator. PatchGAN outperformed the other three popular GANs, DCGAN, WGAN and WGAN-GP, in learning 256×256 pixels meandering fluvial patterns composed of three facies. PatchGAN realisations have more accurate geology than the other three GANs in the comparison study.

3. Using gradient penalty terms in the loss function is a recommended way of preventing mode collapse, which means GANs can only create one or a few samples instead of all types in the training dataset, though this method is also data- and task-depended. The zero-centred gradient penalty greatly increased the generational diversity in Fluvial GAN 2D realisations while performing undesirable when applied to the 3D reconstruction framework, FluvialGAN_3DR. Instead, the one-centred gradient penalty suited the FluvialGAN_3DR, favouring a higher diversity of sandbody type in the upper section of different 3D models.

4. The One-Hot Encoder helps GAN reduce the 'mislabelling' issue, which means GAN create facies in the place of other facies whose encoded value (after data pre-processing) is close to them when learning multi-facies distribution. The 'mislabelling' is not obvious when the facies model has three or fewer facies but gets severe when the number of facies is large, typically seven in this thesis. The One-Hot Encoder is a standard method of encoding categorical data to numeric. Thus, it is more suitable for dealing with the multi-facies model and is highly recommended based on the result of the study in this thesis. Of course, the data pre-processing has to use other algorithms if the number of facies is too big, for example, twenty facies. Because the One-Hot Encoder expands the data size several times equal to the number of facies, making the GAN training more difficult as the target data volume significantly increases.

5. The proposed Hybrid-discriminator composed of a convolution-based discriminator and a dilated convolution-based discriminator helps GAN learn meandering fluvial geology better by reducing the occurrence of geologically unrealistic features, such as 'closed channel'. The 'closed channel' pattern is a loop pattern unseen in the training dataset but exists in many GANs generations studied in this thesis. The 'closed channel' pattern features plausible locally but unrealistic globally, indicating a strong correlation to the receptive field of convolutional neural networks. Using dilation can efficiently increase the receptive field while maintaining the kernel size unchanged in convolution but introduces the 'gridding effect'. A hybrid-dilation discriminator adopting different dilation rates cannot reduce the 'gridding effect' in the case of this thesis. Therefore, the Hybrid-discriminator uses a PatchGAN

discriminator without dilation and a PatchGAN discriminator with a hybrid dilation design to jointly penalise the input data, which reaches a good balance between the 'gridding effect' and geological realism.

6. Using the conditional GAN-based approach to predict an upper slice based on a given lower slice is a more geologically intuitive way of applying GAN to 3D geomodelling in line with conventional geostatistical geomodelling algorithms. The training of this approach requires pairs of data in the form of a target upper slice and a conditioning lower slice, which feeds into the discriminator together by concatenating in the spectral dimension to prevent the generator from duplicating the given lower slice as both slices are real data from the training dataset. Unlike common GAN-based 3D modelling, this conditional GAN-based 3D reconstruction, FluvialGAN_3DR, allows the user to define the target reservoir thickness/number of layers without re-training or post-processing. This bright feature makes the pre-trained GAN models more reusable for building up 3D models, avoiding wasting time re-training them when needed to apply them to a similar reservoir with different thicknesses. Particularly, this approach surpasses fully convolutional-based GANs in controlling the vertical-to-lateral size ratio because fully convolutional-based GANs can only vary their output size with fixed ratios to their latent vector size.

7. The proposed multi-step training enhancement can help preserve the meandering channel geometry and placement of associated facies in the 3D reconstruction process of FluvialGAN_3DR. By involving the reconstruction in training, the multi-step enhancement defines multiple target upper slices that the generator needs to predict by sequentially reconstructing upwards with different steps/times. The training, therefore, no longer just aims to predict the upper slice immediately above the given lower slice but infers the development of meandering channels in a sedimentary section deposited during a period. This enhancement effectively prevents the pre-trained reconstruction process from gradually losing geological realism, which means the predicted upper slices no longer present appropriate meandering fluvial patterns with forward reconstruction continues building up upper layers.

8. The proposed conditioning data decay enhancement improves the conditional GAN-based reconstruction's learning capability in the spatial correlation strength between

slices reflecting the sedimentary settings. This enhancement introduces historical GAN generations into the conditioning data in a decaying manner, imitating vertical correlations among geomodel layers in a purely aggrading system. This conditioning data decay enhancement prevents FluvialGAN_3DR from collapsing to generate high avulsion rate models, enriching the model diversity and converging a good range of sand connectivity.

9. Coupled avulsion rate with the conditioning data decay enhancement introduces an extra parameter, the avulsion rate factor, into the FluvialGAN_3DR framework to affect 3D geological patterns, which heuristically relates the avulsion rate settings of the 3D training data to the spatial correlation strength in a purely aggrading environment. The avulsion rate factor in the conditioning data decay enhancement provides an alternative way of controlling GANs' sense of sedimentary settings. The traditional conditional GAN uses global features as an extra input to the neural networks, hindering the modeller from understanding how it impacts the outputs. In contrast, this approach explicitly incorporates the understanding of geological knowledge into the conditional GAN training and generation, making it easier to understand the global features' impact on GAN generation.

7.2 Recommendations

This thesis is a step forward in deep generative models application to geomodelling from learning object-based to process-based simulations while leaving a lot of unsolved issues for further investigations. The author highly recommends researchers who are interested in this field consider the perspectives below as potential future studies:

1. How to reduce the latent vector size of GAN while not undermining the generation quality needs a thoughtful study of the relationship between the generator's input and the output size. One potential use of GANs is incorporating the pre-trained generator into the history-matching workflow to update reservoir models. To reduce the computational cost and enable the use of heuristic-based optimisers, e.g. particle swarm optimiser, the history matching workflow would better have as few parameters as

possible. The latent vector of GAN will introduce input parameters into the model updating loop if the pre-trained generator is involved in the workflow. Therefore, extending GANs applications to other pipelines would benefit from a smaller latent vector. However, according to the preliminary study in Section 6.1.1, decreasing the latent vector size worsens Fluvial GAN's generation quality. A hyper-parameter optimisation study on GAN's structure, particularly the generator's architecture, would benefit the integration of GANs and the model updating loop.

2. Rethink the use of GAN to simulate 3D models would enable the user to control the 3D models' size, which allows the reuse of the pre-trained GAN, saving considerable time on dataset preparation and GAN training. When different reservoirs have similar sedimentary environments but contrasting sizes, GANs have to run the training again with a new model structure and even need a new training dataset because they are inflexible with their output size. This thesis proposed FluvialGAN_3DR that allows modelling a deposit of an arbitrary thickness but can not customise the lateral extent. To the best scope known, current methods still rely on increasing some layers' size of a pre-trained model to proportionally change the size [Laloy et al., 2018, Song et al., 2022]. However, based on the preliminary result in Section 6.1.2, expanding the latent vector in the spatial dimension can enlarge Fluvial GAN's generation size proportionally but cause realism losses in both geological patterns and sand connectivity. Also, this method increases the latent vector size, which is unfavourable considering the potential use of GANs in model updating. How to freely define GAN generations' lateral extent size and even customise the field shape remain a problem to be solved.

3. Applying data conditioning to GAN realisations still needs more effort to bridge GAN learning and data conditioning harmonically and efficiently. Current methods, to the best scope of the author's knowledge, either require post-processing to condition GAN generations to observed data or directly use conditional GAN to infer facies models based on given observed data. As discussed in Section 6.2, both approaches have natural drawbacks impeding them from becoming a general solution. For the post-processing-based framework, though they largely preserve the learned patterns and allow the reuse of a pre-trained GAN, the extra optimisations either have

difficulty in converging and even fail to converge or highly rely on the initialisation of the latent vector, which is time-consuming and cannot satisfy the demanding of hard conditioning, e.g. well data, On the other hand, conditional GAN performs soft conditioning and accepts nearly all kinds of data as an input, which makes particularly seismic conditioning a low-hanging fruit compared to well data that requires hard conditioning. In Chapter 5, SPADE-based FluvialGAN_3DR uses historical generations as the conditioning data to condition the generated upper slice successfully. While Park et al. [2019] used more semantic classes (e.g. 35, 150, 182 from different datasets) to condition their SPADE generator than this study (7 maps). So, it is possible to condition Fluvial GAN 3D generations by adding more maps, e.g. seismic amplitude or impedance maps to the conditioning data (input of SPADE). One clear drawback, as discussed in Appendix A, is that current conditional GAN-based frameworks train GAN to learn geological patterns and condition observed data synchronously. This training way greatly reduces the chance of reusing pre-trained GAN models. This thesis tried to separate the conditional GAN training into GAN learning and data conditioning in Appendix A. However, this preliminary study results in undesirable generation quality, which may need to redesign the way of introducing conditioning data into the framework. The idea of making the GAN application a more general tool is to separate GAN learning and data conditioning. So that the modeller can reuse the learned geological patterns and adapt them to the observed data available in the target reservoirs.

4. GAN evaluation is another urgent and crucial topic in the GAN application field, not only for facies modelling. Data containing complex patterns, e.g. images and facies models, raises quality evaluation challenges that quantitative scores cannot perfectly describe and replace human/expert interpretation, such as photo-realism in the computer vision domain, geological realism in geomodelling, etc. Computer scientists leave this challenge to neural networks, which is what GAN is doing, using a neural network, the discriminator, to judge if a sample is realistic. Though this eased the pain of evaluating complex data, who can judge if the GANs do a proper job? In the past decades, many indicators, such as net-to-gross, connectivity, etc., provided quantitative descriptions of geological models and revealed key geological features

considered in reservoir modelling. However, those indicators cannot fully represent the geological realism that geologists often need to interpret based on their knowledge of conceptual geology and outcrops. In photosynthesis studies, researchers often use pre-trained image classifiers to evaluate GAN generation quality; for example, the inception score evaluates GANs by calculating the relative probability of each random GAN generation's label and the diversity of predicted labels of the ensemble of random generations [Salimans et al., 2016]. Those scores have two significant questions to answer. First, how to get a pre-trained classifier and associated labelled dataset used for geomodelling that convinces geologists and data scientists? Second, neural networks are still black-box models, and GAN itself uses one neural network (discriminator) to supervise the other (generator). How to justify that it is fair to use another neural network to supervise GAN? Inspired by the rapid scene categorization method [Borji, 2019, Goodfellow et al., 2020] in computer vision, this thesis used an occurrence-based qualitative score to evaluate Fluvial GAN 2D realisation in Section 3.2.3, which counts on how many realisations contain a recurrent unrealistic feature. However, this score is very subjective and time-consuming. A rule-based method to replace the manual check with auto-evaluation could be helpful.

5. Making the GAN model and neural networks, in general, more interpretable and explainable is another significant topic. As mentioned earlier, neural networks lack transparency, making understanding and improving GAN models difficult. Once the GAN model becomes more interpretable, geologists will be more confident and convinced with the GAN generations. This thesis discussed the potential use of an extra parameter outside the GAN model to control the correlation strength between conditioning data and target in Section 5.2.5. This parameter has a straightforward definition and is heuristically correlated to the avulsion, a geological process, making user easier to understand its effect on GAN reconstructed 3D models. Of course, this work is insufficient to make GANs transparent, and further deep investigations are needed.

6. How to improve deep generative models' generations quality based on reusing pre-trained (partial) neural networks as the new model's components would trigger the

next boom of their development. Due to the lack of understanding, many GAN improvements rely on extensive explorations of hyper-parameters and architectures or inspired inventions of methods, model structures, and training strategies. GAN variants perform well and badly when given different datasets, which takes a long time to find a suitable configuration. If a framework could keep updating a GAN generator to improve the performance in different aspects by reusing pre-trained neural network models or layers, the GAN-based simulator will deserve long-term development as the computations cost spent on earlier stage GAN learning would not be a waste. This idea shares a similar purpose as transfer learning, where researchers use pre-trained neural networks in one domain (commonly image classification tasks) as the intermediate model in other research areas to tackle different tasks [Bozinovski and Fulgosi, 1976]. So, many unrealistic features identified in both Fluvial GAN and FluvialGAN3D generations could be continuously improved in a shorter time without training everything from the beginning, which may even give a worse result in other aspects.

7. A thoughtful investigation of GAN applications to facies modelling in the presence of uncertainty in sedimentary environment interpretation would enable a further discussion on handling a multiple-scenario challenge. This thesis only focused on low NTG meandering fluvial systems modelled by FLUMY. The Fluvial GAN and FluvialGAN3D may perform differently when applied to high NTG meandering fluvial or other fluvial systems (e.g. braided and anastomosing rivers) and even different sedimentary environments (e.g. delta and deep marine). Two approaches can be seen to implement GANs uncertainty between alternative sedimentary environments. One is to optimise GANs for each sedimentary system and use another control, either an expert's or computer-assisted decision on which GAN should be used to model a reservoir. Another one is to train a GAN using a data-rich dataset containing all possible patterns and then study the GAN latent space to optimise the ranges of each element to discover the best-fit patterns for a reservoir. Both approaches need considerable efforts to exploit in depth.

Appendix A

Conditional GAN-based Approach

This section applies the GAN conditioning technique used in Chapter 5 to Fluvial GAN for 2D seismic conditional facies modelling. Conditional GAN takes additional data, such as associated labels or maps, as an extra input to soft condition the output to those data. A couple of conditional GAN techniques blend conditioning data into the generator, for example, conditional normalisation technique [Park et al., 2019]. Spatially-adaptive denormalisation (SPADE) is one popular conditional normalisation that extracts features from conditioning data to bias the normalisation calculation (see Section 2.3.3.3 for the details) [Park et al., 2019]. This method is also one of the essential elements in the reconstruction-based Fluvial 3D framework (see Section 5.2), where the SPADE condition the target upper layers to the layers below. Indeed, SPADE can also condition GAN generations to seismic data with appropriate data pre-processing. However, as mentioned in section 6.2, the traditional training process of conditional GAN limits the reuse of the pre-trained model when the new conditioning data has different types or ranges from the conditioning data used for training.

This section uses a pre-trained Fluvial GAN as an example to present the preliminary results of decoupling the data conditioning from SPADE-based conditional GAN training. This SPADE-based conditional framework works as a post-processing of Fluvial GAN with a self-training loop (see Figure A.1). This framework consists of a SPADE generator, an external forward modelling tool and a pre-trained Fluvial GAN, including the generator and the discriminator. The pre-trained Fluvial GAN generator with fixed parameters produces unconditional realisations as the target for this conditional framework. The SPADE generator has the same architecture as the Fluvial GAN generator, except for adding the SPADE elements to all batch normalisation layers. The SPADE generator loads the pre-trained Fluvial GAN generator's parameters and freezes those parameters during training. Therefore, only the SPADE elements in this framework have initialised learnable parameters, which will be optimised during training. This feature makes the connection to the external model that contains in-differentiable calculation possible. Any forward modelling tool can integrate into this conditioning framework to calculate the conditioning data the user wants to condition Fluvial GAN generations. This study uses a simplified version of a petro-elastic modelling tool presented in FAHIMUDDIN [2009] to simulate synthetic seismic impedance. The SPADE generator takes the synthetic seismic impedance associated with unconditional realisations as the conditioning data to simulate conditional realisations. The discriminator of the pre-trained Fluvial GAN extracts features from the unconditional and conditional realisations to compute the mean absolute difference as the loss, known as the feature loss [Park et al., 2019, Wang et al., 2018b], to update the SPADE elements' parameters.

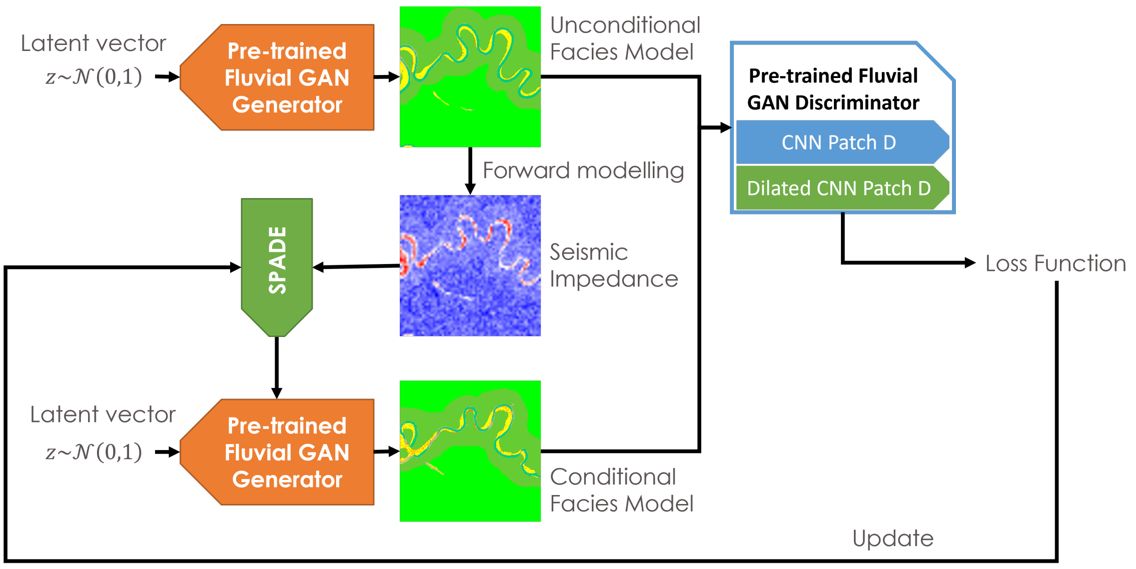

Latent vector
$z \sim \mathcal{N}(0,1)$
Pre-trained Fluvial GAN Generator
Unconditional Facies Model
Pre-trained Fluvial GAN Discriminator
CNN Patch D
Dilated CNN Patch D
Forward modelling
SPADE
Seismic Impedance
Loss Function
Latent vector
$z \sim \mathcal{N}(0,1)$
Pre-trained Fluvial GAN Generator
Conditional Facies Model
Update

FIGURE A.1: SPADE-based conditional Fluvial GAN training workflow.

As the resolution of the facies model is much higher than seismic, this framework linearly upscales the synthetic acoustic impedance before feeding into the SPADE generator. This reduces the spatial size of the conditioning data from 256×256 to 64×64. Considering the grid size of the facies model is 10 meters, one pixel of the upscaled acoustic impedance slice represents a $40m \times 40m$ grid.

Further data pre-processing on the upscaled acoustic impedance is still necessary because its values are beyond the desired input range of common machine-learning algorithms. This study shows an example of a further data transformation that duplicates the acoustic impedance maps twice and centralises the value at 8.5, as its original range is about 7 to 10 GPa. One map minus 8.5, and the other reverses the calculation of the first one. This transformation scales the acoustic impedance map to a smaller range and uses two transformed maps with reversed processes to avoid introducing potential biases from the value to the conditional GAN. The data transformation presented here exemplifies one way of pre-processing the acoustic impedance as a preliminary result. Other methods of transforming data are also encouraged.

The training of this framework demands both CPU and GPU, slowing down the processors' performance in computation speed due to the switch between CPU and GPU calculations. A GPU-based forward modelling tool can significantly accelerate the whole training process. The training for 100,000 iterations takes 24 hours using a single GPU RTX3090 and a single 3.6 GHz CPU Intel(R) Xeon(R) W-2133.

After training, the SPADE generator can produce conditional realisations by feeding a given synthetic acoustic impedance map. As this framework uses random generations from Fluvial GAN to train the SPADE elements, GAN River-I can work as the test set for the SPADE generator. The subset of GAN River-I in section 3.2.2 feeds into the forward modelling simulator to produce 1600 synthetic acoustic impedance slices. Then, the trained SPADE generator creates conditional facies realisations based on those acoustic impedance slices.

The conditional Fluvial GAN infers the conditional facies realisations, overlapping significantly with the ground truth regarding the sand proportion and connectivity range and in the UMAP visualisation. Compared to the ground truth, conditional Fluvial GAN produces less low connectivity samples with NTG above 0.3, slightly underestimating the connectivity uncertainty (see Figure A.2 a). On the other hand, the conditional Fluvial GAN faithfully creates channel elements based on the given acoustic impedance map, showing a pretty high coincidence to the ground truth in UMAP visualisation (see Figure A.2 b). This is because the acoustic impedance map provides the location of geobodies to the conditional generator, resulting in the conditional facies models locating geobodies at the same place as the training data.

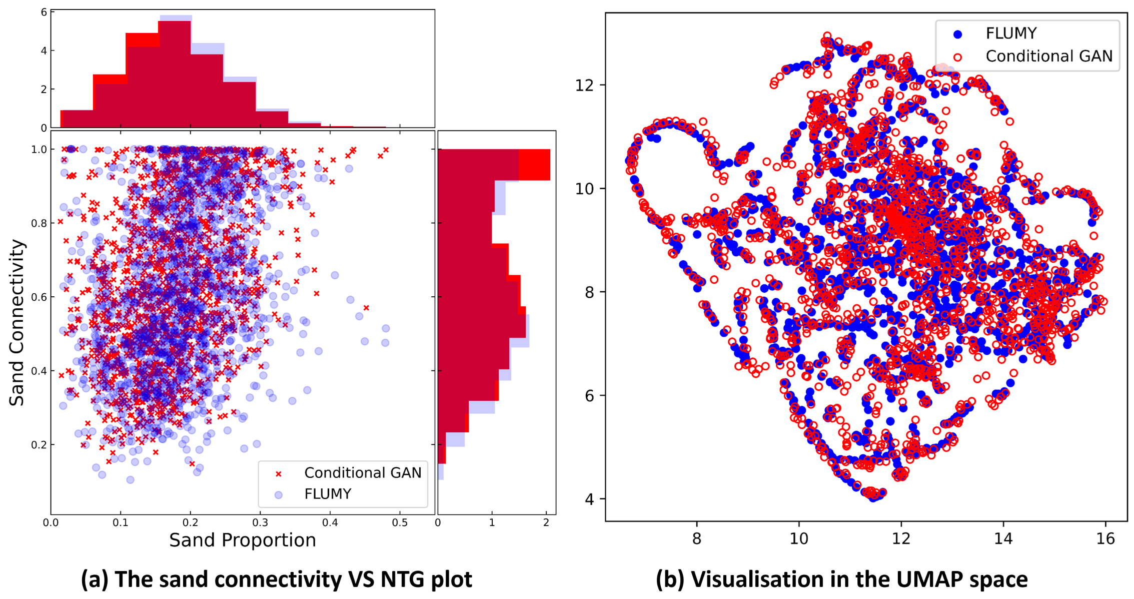


Figure A.2: The sand connectivity against proportion plot and UMAP visualisation of conditional realisations from the SPADE generator. Blue points are FLUMY realisations. Red points are SPADE-based conditional GAN realisations.

However, realisations from the SPADE-based conditional Fluvial GAN are not visually competitive to the slices from GAN River-I, though they honour the meandering shape and local spatial correlations (see Figure A.3). The SPADE-based conditional Fluvial GAN preserves the sinuous shape of channels and local facies transition from channel centrelines to distal, which is what the pre-trained Fluvial GAN learned from GAN River-I. Unlike the unconditional realisations from the Fluvial GAN or the training data from GAN River-I, conditional realisations are more likely to contain broken features and tend to miss some small thin channels.

A further analysis comparing the Fluvial GAN's generations with and without conditioning data reveals that the decrease in the reproduction quality may result from a natural flaw of this conditioning framework design. The latent vector of Fluvial GAN samples facies realisations in a parametric manner, which means its values control the fluvial elements' pattern and location. While the conditioning framework still initialises the latent vector stochastically. This causes the conflict of the sand bodies' location between information

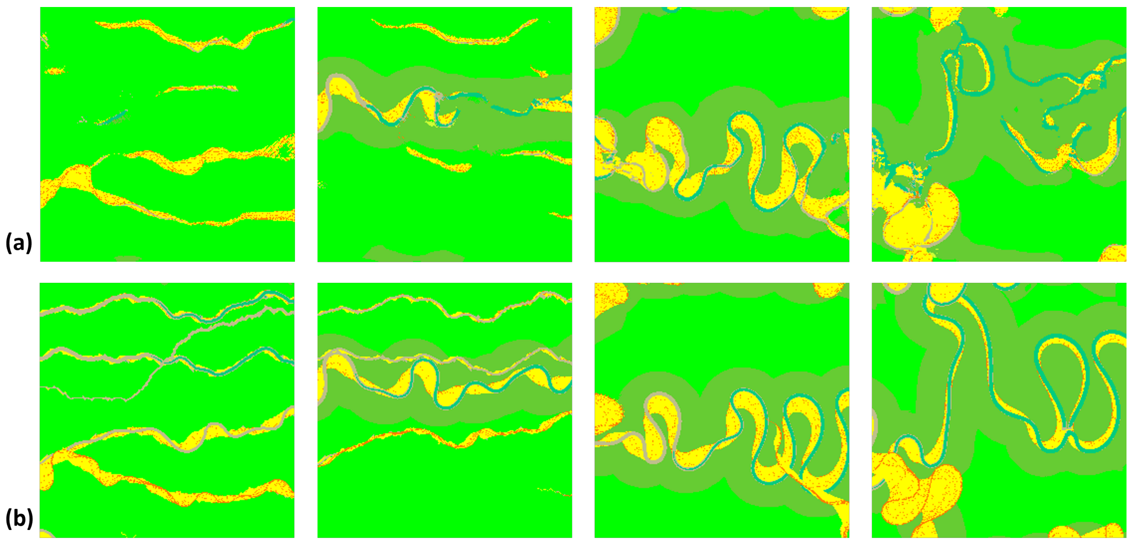


FIGURE A.3: Examples of conditional Fluvial GAN realisations and their counterparts in GAN River-I. (a) different realisations from conditional Fluvial GAN. (b) the ground truth of the facies models producing the acoustic impedance maps for conditional Fluvial GAN

from the latent vector and acoustic impedance map. An example illustrates this conflict of location information from the two sources (see Figure A.4). The pre-trained Fluvial GAN produce an unconditional realisation with a sand body at the bottom of the image by inputting a latent vector (see Figure A.4 a). However, the acoustic impedance map indicates the upside of the image contains a sand body, which contradicts the unconditional realisation. So, when the Fluvial GAN, given this latent vector, uses this acoustic impedance map as the conditioning data, the SPADE has to assign big weights to its parameters to bias the neural networks in Fluvial GAN. This information conflict, therefore, undermines the generation quality of conditional Fluvial GAN.

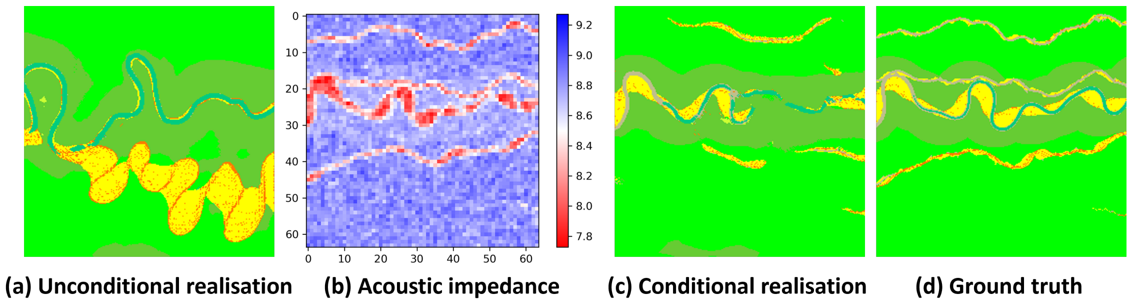


FIGURE A.4: An example of the stochastic nature Fluvial GAN unconditional generation conflicting the conditioning data from the acoustic impedance map. (a) an unconditional realisation from Fluvial GAN given a latent vector. (b) the acoustic impedance of a 2D slice from GAN River-I. (c) a conditional realisation from SPADE-based Fluvial GAN using the same latent vector. (d) the ground truth of the 2D slice.

Another trial on the generation's diversity reveals a limited variance of realisations conditioned to the same acoustic impedance map when feeding different input vectors. Figure A.5 shows an example of varying input vector results in limited changes of the conditional realisation. Only the mud plug facies downstream present different broken levels, changing the facies' connectivity. This low variance further proves the SPADE elements dominate the forward calculation of the Fluvial GAN, leading to the ineffectiveness of the latent vector.

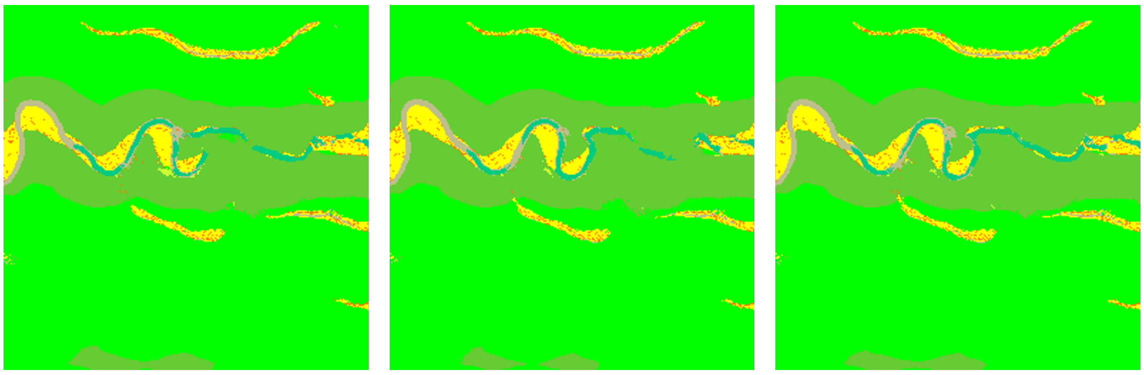


FIGURE A.5: Conditional realisations conditioned to the same acoustic impedance map using different latent vectors as the input.

Therefore, this preliminary experiment demonstrates that the SPADE-based conditioning framework can blend conditioning data into Fluvial GAN generation without retraining the whole Fluvial GAN model but needs further development to balance the strength of conditioning data. Three potential ways of improving this framework are: (1) modify the loss function to reduce the impact of conditioning data on facies realisations; (2) further blur the acoustic impedance or compute the seismic amplitude map if fairly assumptions of layers surrounding it are available as even the upscaled one still shows very clear sand bodies' shapes; (3) rethink the way of using SPADE to merge conditioning data into the pre-trained Fluvial GAN model.

This section extends the proposed conditioning framework to seismic soft conditioning in principle. Still, the full 3D conditioning GAN study is outside this thesis and can be seen as a further seamless extension of this work. The approach presented in Chapter 5 deals with 2D facies models iteratively, making it practical to integrate the conditioning method applied to 2D images into the iterative approach, for example, conditional normalisations, e.g. SPADE. Adding more maps, e.g. seismic impedance maps, to the conditioning data as the input of SPADE is a possible method of conditioning Fluvial GAN 3D simulations, though foreseeable CPU/GPU demanding increases have to be tackled.

References

A. Araujo, W. Norris, and J. Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019. doi: 10.23915/distill.00021. https://distill.pub/2019/computing-receptive-fields.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

L. Azevedo, G. Paneiro, A. Santos, and A. Soares. Generative adversarial network as a stochastic subsurface model reconstruction. *Computational Geosciences*, 24(4):1673–1692, 2020.

J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

P. W. Bogaart, R. Van Balen, C. Kasse, and J. Vandenberghe. Process-based modelling of fluvial system response to rapid climate change—i: model formulation and generic applications. *Quaternary Science Reviews*, 22(20):2077–2095, 2003.

A. Borji. Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65, 2019.

G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

S. Bozinovski and A. Fulgosi. The influence of pattern similarity and transfer learning upon training of a base perceptron b2. In *Proceedings of Symposium Informatica*, volume 3, pages 121–126, 1976.

J. S. Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer, 1990.

A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

A. Bubnova. *On the conditioning of process-based channelized meandering reservoir models on well data*. PhD thesis, Université Paris sciences et lettres, 2018.

C. E. Burns, N. Mountney, D. Hodgson, and L. Colombera. Anatomy and dimensions of fluvial crevasse-splay deposits: Examples from the cretaceous castlegate sandstone and neslen formation, utah, usa. *Sedimentary Geology*, 351:21–35, 2017.

J. Caers. *Modeling Uncertainty in the Earth Sciences*. Wiley Online Library, 2011.

S. A. Canchumuni, A. A. Emerick, and M. A. Pacheco. Integration of ensemble data assimilation and deep learning for history matching facies models. In *OTC Brasil*. OnePetro, 2017.

S. Chan and A. H. Elsheikh. Parametric generation of conditional geological realizations using generative neural networks. *Computational Geosciences*, 23(5):925–952, 2019.

O. Chapelle and M. Wu. Gradient descent optimization of smoothed information retrieval metrics. *Information retrieval*, 13:216–235, 2010.

I. Cojan, O. Fouché, S. Lopéz, and J. Rivoirard. Process-based reservoir modelling in the example of meandering channel. In *Geostatistics Banff 2004*, pages 611–619. Springer, 2005.

P. W. Corbett. The role of geoengineering in field development. *New Technologies in the Oil and Gas Industry*, 2012.

L. Datta. A survey on activation functions and their relation with xavier and he normal initialization. *arXiv preprint arXiv:2004.06632*, 2020.

A. J.-C. de Saint-Venant et al. Théorie du mouvement non-permanent des eaux, avec application aux crues des rivières et à l' introduction des marées dans leur lit. *CR Acad. Sci. Paris*, 73(147-154):237–240, 1871.

C. V. Deutsch. Fortran programs for calculating connectivity of three-dimensional numerical models and for ranking multiple realizations. *Computers & Geosciences*, 24 (1):69–76, 1998.

C. V. Deutsch and L. Wang. Hierarchical object-based stochastic modeling of fluvial reservoirs. *Mathematical geology*, 28(7):857–880, 1996.

C. V. Deutsch, A. G. Journel, et al. Geostatistical software library and user’s guide. *Oxford University Press*, 8(91):0–1, 1992.

M. E. Donselaar and I. Overeem. Connectivity of fluvial point-bar deposits: An example from the miocene huesca fluvial fan, ebro basin, spain. *AAPG bulletin*, 92(9):1109–1129, 2008.

V. Dumoulin and F. Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

F. Ethridge and S. Schumm. Fluvial seismic geomorphology: a view from the surface. *Geological Society, London, Special Publications*, 277(1):205–222, 2007.

A. FAHIMUDDIN. Petro-elastic modeling of a north sea reservoir: Rock physics recipe and eclipse simulator. 2009.

C. Fan, M. Chen, X. Wang, J. Wang, and B. Huang. A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. *Frontiers in Energy Research*, 9:652801, 2021.

R. Ferguson. The threshold between meandering and braiding. In *Channels and Channel Control Structures: Proceedings of the 1st International Conference on Hydraulic Design in Water Resources Engineering: Channels and Channel Control Structures, University of Southampton, April 1984*, pages 749–763. Springer, 1984.

C. R. Ferring. Rates of fluvial sedimentation: implications for archaeological variability. *Geoarchaeology*, 1(3):259–274, 1986.

P. A. Gagniuc. *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017.

H. Gholamalinezhad and H. Khosravi. Pooling methods in deep neural networks, a review. *arXiv preprint arXiv:2009.07485*, 2020.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11): 139–144, 2020.

J.-L. Grimaud, F. Ors, M. Lemay, I. Cojan, and J. Rivoirard. Preservation and completeness of fluvial meandering deposits influenced by channel motions and overbank sedimentation. *Journal of Geophysical Research: Earth Surface*, page e2021JF006435, 2022.

F. B. Guardiano and R. M. Srivastava. Multivariate geostatistics: beyond bivariate moments. In *Geostatistics Troia' 92*, pages 133–144. Springer, 1993.

D. R. Guichard et al. *Single and Multivariable Calculus*. David R. Guichard, 2020.

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

J. Han and C. Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *International workshop on artificial neural networks*, pages 195–201. Springer, 1995.

J. Han, J. Pei, and H. Tong. *Data mining: concepts and techniques*. Morgan kaufmann, 2022.

J. T. Hancock and T. M. Khoshgoftaar. Survey on categorical data for neural networks. *Journal of Big Data*, 7(1):1–41, 2020.

S. Hanson and L. Pratt. Comparing biases for minimal network construction with back-propagation. *Advances in neural information processing systems*, 1, 1988.

R. Hauge, L. Holden, and A. R. Syversveen. Well conditioning in object models. *Mathematical geology*, 39(4):383–398, 2007.

H. Hayashi, K. Abe, and S. Uchida. Glyphgan: Style-consistent font generation based on generative adversarial networks. *Knowledge-Based Systems*, 186:104927, 2019.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

P. L. Heller and C. Paola. Downstream changes in alluvial architecture; an exploration of controls on channel-stacking patterns. *Journal of Sedimentary Research*, 66(2):297–306, 1996.

E. Hoogeboom, A. A. Gritsenko, J. Bastings, B. Poole, R. v. d. Berg, and T. Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.

J. M. Hovadik and D. K. Larue. Static characterizations of reservoirs: refining the concepts of connectivity and continuity. *Petroleum Geoscience*, 13(3):195–211, 2007.

S. Ikeda, G. Parker, and K. Sawai. Bend theory of river meanders. part 1. linear development. *Journal of Fluid Mechanics*, 112:363–377, 1981.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

M. M. Islam and J.-M. Kim. Vision-based autonomous crack detection of concrete structures using a fully convolutional encoder–decoder network. *Sensors*, 19(19):4251, 2019.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

N. Jetchev, U. Bergmann, and R. Vollgraf. Texture synthesis with spatial generative adversarial networks. *arXiv preprint arXiv:1611.08207*, 2016.

H. Johannesson and G. Parker. Linear theory of river meanders. *River meandering*, 12: 181–213, 1989.

T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pages 1942–1948. IEEE, 1995.

Y. Kim, S. Keely, J. Ghosh, and H. Ling. Application of artificial neural networks to broadband antenna design based on a parametric frequency model. *IEEE Transactions on Antennas and Propagation*, 55(3):669–674, 2007.

P. King. The connectivity and conductivity of overlapping sand bodies. In *North sea oil and gas reservoirs—II*, pages 353–362. Springer, 1990.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

E. Laloy, R. Hérault, J. Lee, D. Jacques, and N. Linde. Inversion using a new low-dimensional representation of complex binary geological media based on a deep neural network. *Advances in water resources*, 110:387–405, 2017.

E. Laloy, R. Hérault, D. Jacques, and N. Linde. Training-image based geostatistical inversion using a spatial generative adversarial neural network. *Water Resources Research*, 54(1):381–406, 2018.

D. K. Larue and J. Hovadik. Connectivity of channelized reservoirs: a modelling approach. *Petroleum Geoscience*, 12(4):291–308, 2006.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

C. Li and M. Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European conference on computer vision*, pages 702–716. Springer, 2016.

J. H. Lim and J. C. Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

H. Lin, Z. Shi, and Z. Zou. Maritime semantic labeling of optical remote sensing images with multi-scale fully convolutional network. *Remote sensing*, 9(5):480, 2017.

W. A. Little. The existence of persistent states in the brain. *Mathematical biosciences*, 19 (1-2):101–120, 1974.

Y. Liu, A. Harding, R. Gilbert, and A. G. Journel. A workflow for multiple-point geostatistical simulation. In *Geostatistics Banff 2004*, pages 245–254. Springer, 2005.

Y. Liu, W. Sun, and L. J. Durlofsky. A deep-learning-based geological parameterization for history matching complex models. *Mathematical Geosciences*, 51(6):725–766, 2019.

S. Lopez. *Modélisation de réservoirs chenalisés méandriformes: une approche génétique et stochastique*. PhD thesis, École Nationale Supérieure des Mines de Paris, 2003.

S. Lopez, I. Cojan, J. Rivoirard, and A. Galli. Process-based stochastic modelling: meandering channelized reservoirs. *Analogue and Numerical Modelling of Sedimentary Systems: From Understanding to Prediction, Wiley, Oxford, UK*, pages 139–144, 2009.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

A. L. Maas, A. Y. Hannun, A. Y. Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, Georgia, USA, 2013.

A. Maharaja. Tigenerator: object-based training image generator. *Computers & Geosciences*, 34(12):1753–1761, 2008.

B. Makaske. Anastomosing rivers: a review of their classification, origin and sedimentary products. *Earth-Science Reviews*, 53(3-4):149–196, 2001.

S. Mandelli, V. Lipari, P. Bestagini, and S. Tubaro. Interpolation and denoising of seismic data using convolutional neural networks. *arXiv preprint arXiv:1901.07927*, 2019.

G. Mariethoz and J. Caers. *Multiple-point geostatistics: stochastic modeling with training images*. John Wiley & Sons, 2014.

G. Mariethoz, P. Renard, F. Cornaton, and O. Jaquet. Truncated plurigaussian simulations to characterize aquifer heterogeneity. *Groundwater*, 47(1):13–24, 2009.

M. Masihi and P. R. King. Percolation approach in underground reservoir modeling. *Water Resources Management and Modeling*, 2012.

L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

T. McKie and P. Audretsch. Depositional and structural controls on triassic reservoir performance in the heron cluster, etap, central north sea. In *Geological Society, London, Petroleum Geology Conference series*, volume 6, pages 285–297. Geological Society of London, 2005.

J. Menick and N. Kalchbrenner. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. *arXiv preprint arXiv:1812.01608*, 2018.

L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

A. D. Miall. A review of the braided-river depositional environment. *Earth-Science Reviews*, 13(1):1–62, 1977.

A. D. Miall and A. D. Miall. Facies models. *Stratigraphy: a modern synthesis*, pages 161–214, 2016.

Z. Michalewicz and M. Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary computation*, 4(1):1–32, 1996.

M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

A. Nicholas and D. Walling. The significance of particle aggregation in the overbank deposition of suspended sediment on river floodplains. *Journal of Hydrology*, 186(1-4):275–293, 1996.

A. Niculescu-Mizil, C. Perlich, G. Swirszcz, V. Sindhwani, Y. Liu, P. Melville, D. Wang, J. Xiao, J. Hu, M. Singh, et al. Winning the kdd cup orange challenge with ensemble selection. In *KDD-Cup 2009 Competition*, pages 23–34. PMLR, 2009.

J. H. Nienhuis, T. E. Törnqvist, and C. R. Esposito. Crevasse splays versus avulsions: A recipe for land building with levee breaches. *Geophysical Research Letters*, 45(9): 4058–4067, 2018.

A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016.

T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019.

G. Parker, Y. Shimizu, G. Wilkerson, E. C. Eke, J. D. Abad, J. Lauer, C. Paola, W. E. Dietrich, and V. Voller. A new framework for modeling the migration of meandering rivers. *Earth Surface Processes and Landforms*, 36(1):70–86, 2011.

G. Pirot, R. Joshi, J. Giraud, M. D. Lindsay, and M. W. Jessell. loopui-0.1: indicators to support needs and practices in 3d geological modelling uncertainty quantification. *Geoscientific Model Development*, 15(12):4689–4708, 2022.

M. J. Pyrcz and C. V. Deutsch. *Geostatistical reservoir modeling*. Oxford university press, 2014.

N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

B. Ramsundar and R. B. Zadeh. *TensorFlow for deep learning: from linear regression to reinforcement learning*. " O'Reilly Media, Inc.", 2018.

H. G. Reading. *Facies models*, pages 434–444. Springer Berlin Heidelberg, Berlin, Heidelberg, 1978. ISBN 978-3-540-31079-2. doi: 10.1007/3-540-31079-7_81. URL `https://doi.org/10.1007/3-540-31079-7_81`.

N. Remy, A. Boucher, and J. Wu. *Applied geostatistics with SGeMS: A user's guide*. Cambridge University Press, 2009.

P. Renard and D. Allard. Connectivity metrics for subsurface flow and transport. *Advances in Water Resources*, 51:168–196, 2013.

H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

G. Rongier, P. Collon, P. Renard, J. Straubhaar, and J. Sausse. Comparing connected structures in ensemble of random fields. *Advances in Water Resources*, 96:145–169, 2016.

S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

C. E. Russell. *Prediction of sedimentary architecture and lithological heterogeneity in fluvial point-bar deposits*. PhD thesis, University of Leeds, 2017.

T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

R. T. Saucier. *Geomorphology and Quaternary geologic history of the Lower Mississippi Valley*, volume 1. US Army Engineer Waterways Experiment Station, 1994.

S. A. Schumm. *River variability and complexity*. Cambridge University Press, 2007.

R. Slingerland and N. D. Smith. River avulsions and their deposits. *Annu. Rev. Earth Planet. Sci.*, 32:257–285, 2004.

S. Song, T. Mukerji, and J. Hou. Bridging the gap between geophysics and geology with generative adversarial networks. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11, 2021a.

S. Song, T. Mukerji, and J. Hou. Geological facies modeling based on progressive growing of generative adversarial networks (gans). *Computational Geosciences*, 25(3):1251–1273, 2021b.

S. Song, T. Mukerji, J. Hou, D. Zhang, and X. Lyu. Gansim-3d for conditional geomodeling: Theory and field application. *Water Resources Research*, 58(7):e2021WR031865, 2022.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

S. Strebelle. Multiple-point statistics simulation models: pretty pictures or decision-making tools? *Mathematical Geosciences*, 53(2):267–278, 2021.

S. Strebelle, K. Payrazyan, and J. Caers. Modeling of a deepwater turbidite reservoir conditional to seismic data using principal component analysis and multiple-point geostatistics. *Spe Journal*, 8(03):227–235, 2003.

S. B. Strebelle. *Sequential simulation drawing structures from training images*. Stanford University, 2000.

T. Sun, P. Meakin, and T. Jøssang. Meander migration and the lateral tilting of floodplains. *Water Resources Research*, 37(5):1485–1502, 2001.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

R. Sutton. Two problems with back propagation and other steepest descent learning procedures for networks. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*, pages 823–832, 1986.

T. T. Teoh and Z. Rong. Deep convolutional generative adversarial network. In *Artificial Intelligence with Python*, pages 289–301. Springer, 2022.

H. Thanh-Tung, T. Tran, and S. Venkatesh. Improving generalization and stability of generative adversarial networks. *arXiv preprint arXiv:1902.03984*, 2019.

D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014. ISSN 2167-8359. doi: 10.7717/peerj.453. URL `https://doi.org/10.7717/peerj.453`.

R. G. Walker. Facies models 1. general introduction. *Geoscience Canada*, 3(1):21–24, 1976.

R. G. Walter. *Facies models*. Number 551.7 FAC. 1984.

P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell. Understanding convolution for semantic segmentation. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 1451–1460. Ieee, 2018a.

T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018b.

F. Wellmann and G. Caumon. 3-d structural geological models: Concepts, methods, and uncertainties. In *Advances in Geophysics*, volume 59, pages 1–121. Elsevier, 2018.

C. J. Willems, H. M. Nick, M. E. Donselaar, G. J. Weltje, and D. F. Bruhn. On the connectivity anisotropy in fluvial hot sedimentary aquifers and its influence on geothermal doublet performance. *Geothermics*, 65:222–233, 2017.

A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The marginal value of adaptive gradient methods in machine learning. *Advances in neural information processing systems*, 30, 2017.

Y. Wu and K. He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

Z. Yang, Q. Chen, Z. Cui, G. Liu, S. Dong, and Y. Tian. Automatic reconstruction method of 3d geological models based on deep convolutional generative adversarial networks. *Computational Geosciences*, pages 1–16, 2022.

J. M. Yarus, R. L. Chambers, M. Maucec, and G. Shi. Facies simulation in practice: Lithotype proportion mapping and plurigaussian simulation, a powerful combination. In *Paper P-014 presented at the 9th International Geostatistics Congress, Oslo, Norway*, pages 11–15, 2012.

F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

C. Zhang, X. Song, and L. Azevedo. U-net generative adversarial network for subsurface facies modeling. *Computational Geosciences*, 25(1):553–573, 2021.

H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019a.

T.-F. Zhang, P. Tilke, E. Dupont, L.-C. Zhu, L. Liang, and W. Bailey. Generating geologically realistic 3d reservoir facies models using deep learning of sedimentary architecture with generative adversarial networks. *Petroleum Science*, 16(3):541–549, 2019b.