Universidade Estadual de Campinas
Instituto de Computação

Jonlenes Silva de Castro

Redes Adversárias Generativas para Remoção de Ruídos em Dados Sísmicos

CAMPINAS
2020

Jonlenes Silva de Castro

Redes Adversárias Generativas para Remoção de Ruídos em Dados Sísmicos

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Orientadora: Profa. Dra. Sandra Eliza Fontes de Avila
Coorientador: Dr. Tiago Antonio Alves Coimbra

Este exemplar corresponde à versão final da Dissertação defendida por Jonlenes Silva de Castro e orientada pela Profa. Dra. Sandra Eliza Fontes de Avila.

CAMPINAS
2020

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

C279r

Castro, Jonlenes Silva de, 1996-
Redes adversárias generativas para remoção de ruídos em dados sísmicos / Jonlenes Silva de Castro. – Campinas, SP : [s.n.], 2020.

Orientador: Sandra Eliza Fontes de Avila.
Coorientador: Tiago Antonio Alves Coimbra.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Aprendizado profundo. 2. Redes adversárias generativas. I. Avila, Sandra Eliza Fontes de, 1982-. II. Coimbra, Tiago Antonio Alves, 1981-. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Generative adversarial networks for noise removal in seismic data
Palavras-chave em inglês:
Deep learning
Generative adversarial networks
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Sandra Eliza Fontes de Avila [Orientador]
Lúcio Tunes dos Santos
Hélio Pedrini
Data de defesa: 03-08-2020
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0002-7502-650X
- Currículo Lattes do autor: http://lattes.cnpq.br/7242698732966171

Universidade Estadual de Campinas
Instituto de Computação

Jonlenes Silva de Castro

Redes Adversárias Generativas para Remoção de Ruídos em Dados Sísmicos

Banca Examinadora:

- Profa. Dra. Sandra Eliza Fontes de Avila
  Instituto de Computação – UNICAMP
- Prof. Dr. Lúcio Tunes dos Santos
  Instituto de Matemática, Estatística e Computação Científica – UNICAMP
- Prof. Dr. Hélio Pedrini
  Instituto de Computação – UNICAMP

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Campinas, 03 de agosto de 2020

Agradecimentos

Minha imensa gratidão a todos que contribuíram, direta ou indiretamente, para este mestrado.

Agradeço à empresa Quinto Andar, pela bolsa de pesquisa concedida, assim como, à Petrobras pela bolsa complementar. O presente trabalho também foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Agradeço a todos os professores que sempre me incentivaram, em especial meus orientadores Profa. Dra. Sandra Avila e o Prof. Dr. Tiago Coimbra por todo suporte, confiança e parceria durante este período. Agradeço também aos meus colegas dos laboratórios High-Performance Geophysics (HPG) e Reasoning for Complex Data (RECOD) pelo apoio.

Grato a todos os amigos que fizeram parte desta trajetória, os de longa data e aqueles que tive o privilégio de conhecer durante esta jornada, em especial o Welverton que esteve presente desde o primeiro momento desta jornada.

Agradeço aos meus pais, Jonas e Neulice, e minha irmã Janes por sempre estarem apostos para me ajudar a atingir meus objetivos. Por fim, a Camila e a Yasmin, que conviveram comigo durante todo este período, nos dias fáceis e difíceis, celebrando as conquistas e me motivando em todos os momentos.

Resumo

Dados sísmicos fornecem informações das características do ambiente geológico, que, após serem processados, são utilizados na interpretação para identificar e localizar as características do subsolo. O nível de complexidade desta tarefa é diretamente influenciado pela presença de ruídos, artefatos e/ou imprecisões em dados sísmicos. Logo, a remoção de ruídos e o aumento da qualidade destes dados são etapas essenciais no processamento sísmico. Para atenuar estes problemas, propomos abordagens baseadas em redes adversárias generativas (*generative adversarial networks*, GANs) para melhorar os dados sísmicos através da remoção de ruídos e aumento de altas frequências nos dados. Nesse contexto, mapeamos o problema de remoção de ruído como um problema de tradução de imagem para imagem. Ou seja, utilizamos as GANs para aprender a mapear dados sísmicos com ruídos em dados sem ruídos, mantendo a integridade dos dados. Entretanto, o treinamento de redes neurais profundas (como as GANs) tem um problema inerente de escassez de dados rotulados. Para resolver esse problema, desenvolvemos um gerador de dados sísmicos sintéticos rotulados. Com esses dados, criamos nossos modelos baseados nas arquiteturas da Pix2Pix, Pix2PixHD e SPADE e utilizamos estes modelos para realizar a predição de dados sísmicos sintéticos e dados sísmicos reais. Nossos resultados, tanto em dados sísmicos sintéticos quanto em reais, sugerem que as abordagens propostas são promissoras, apresentando melhorias significativas na relação sinal-ruído e no ganho de altas frequências. A acurácia do nosso melhor modelo atingiu aproximadamente 87%, resultado considerado satisfatório para a utilização em cenários reais, de acordo com os especialistas. Com isso, é possível utilizar as GANs como ferramentas de apoio para os especialistas de sísmica, de modo a auxiliar as análises sísmicas e tomadas de decisões de forma mais rápida e assertiva.

Abstract

Seismic data provide information on the geological environment's characteristics, which, after being processed, are used in the interpretation to identify and locate the subsurface characteristics. The complexity of this task is influenced by noise, artifacts, and inaccuracies in seismic data. Therefore, removing noise and increasing the quality of this data are essential steps in seismic processing. To mitigate these problems, we propose approaches based on generative adversarial networks (GANs) to improve seismic data by removing noise and increasing high frequencies in the data. In this context, we mapped the problem of noise removal as an image-to-image translation. We use GANs to learn how to map seismic noise data into data without noise, maintaining the integrity of the data. However, training deep neural networks (such as GANs) has an inherent problem of scarcity of labeled data. To solve this problem, we developed a labeled synthetic seismic data generator. With this data, we created our models based on Pix2Pix, Pix2PixHD and SPADE, and we used these models to perform the prediction of synthetic seismic data and real seismic data. Our results, both in synthetic and real seismic data, suggest that the proposed approaches are promising, showing significant improvements in the signal-to-noise ratio and the gain of high frequencies. The accuracy of our best model achieved about 87% and, according to specialists, this result is considered satisfactory for use in real scenarios. Thus, it is possible to use GANs as support tools for seismic specialists to assist in seismic analysis and decision-making faster and more assertively.

Lista de Figuras

Lista de Tabelas

Lista de Abreviações e Siglas

ACC	Acurácia
cGAN	conditional GAN
CMP	*Common Midpoint*
CNNs	*Convolutional Neural Networks*
CPU	*Central Processing Unit*
DCGAN	*Deep Convolutional* GAN
DCNN	*Deep Convolutional Neural Network*
FFT	*Fast Fourier Transform*
GANAN	GAN Noise Attenuation Network
GANs	*Generative Adversarial Networks*
GD1	Gerador de Dados V1
GD2	Gerador de Dados V2
GPU	*Graphics Processing Unit*
MLP	*Multilayer Perceptrons*
MSE	*Mean Squared Error*
MSSSIM	*Multi-Scale Structural Similarity*
PSF	*Point Spread Function*
PSNR	*Peak Signal-to-Noise Ratio*
SIM	Função de Similaridade
SNR	*Signal-to-Noise Ratio*
SSIM	*Structural Similarity Index*

Sumário

Capítulo 1

Introdução

A busca por depósitos em subsuperfície com potencial econômico de petróleo bruto, gás natural e outros minerais é realizada por vários métodos físicos indiretos. Os principais métodos são os gravimétricos, magnéticos, eletromagnéticos e sísmicos. Esta dissertação tem por objetivo aplicar técnicas de aprendizado de máquina na área de sísmica de exploração, mais precisamente na sísmica de reflexão.

Define-se por *sísmica de reflexão* o método de prospecção geofísica que utiliza os princípios da sismologia para estimar as propriedades da subsuperfície da Terra com base nas reflexões das ondas sísmicas [67]. Sumarizando, a análise dos dados da sísmica de reflexão é dividida em três partes, a saber [78]:

(i) O *processamento sísmico* é desenvolvido para aumentar a razão sinal-ruído e obter informações físicas da região de interesse para habilitar uma interpretação dos dados. Nesta etapa são realizados tratamentos sob os dados adquiridos, em que técnicas, por exemplo, como as de filtragem e imageamento sísmico são utilizadas com a finalidade de prover dados mais representativos e assim melhorar as informações que serão extraídas durante a interpretação destes dados.

(ii) A *inversão sísmica* é o processo de transformar dados da sísmica de reflexão em uma descrição quantitativa das propriedades das rochas de um reservatório, tais como impedância acústica ou de parâmetros elásticos. A inversão sísmica pode ser determinística, aleatória ou geo-estatística. Em resumo, em um conjunto de dados invertido, as amplitudes estão agora descrevendo as propriedades internas das rochas, como tipo litológico, porosidade ou tipo de fluido nas rochas (por exemplo, salmoura ou hidrocarbonetos). Os dados invertidos são ideais para interpretação estratigráfica e caracterização de reservatórios.

(iii) A *interpretação sísmica* é responsável por caracterizar as camadas geológicas em subsuperfície e possibilitar a detecção de estruturas geológicas, tais como o sal, falhas e trapas.

No contexto da redução de incertezas na interpretação sísmica, tem-se que uma imagem sísmica com alta resolução e com uma quantidade mínima de artefatos de processamento é imprescindível. Nos dias atuais, o conhecimento do intérprete no uso de ferramentas

automatizadas é essencial para agilizar e facilitar na interpretação sísmica [11]. No processo de aprimoramento da imagem sísmica, Lomask et al. [38], Stark [65], Wu e Zhang [73], Zeng et al. [80] comentam que a presença de ruídos não coerentes, imprecisões devidos ao um imageamento sísmico de baixa qualidade e artefatos de fenômenos externos aos de interesse são alguns dos fatores que mais influenciam no tempo gasto para interpretar as imagens sísmicas, sendo também um dos principais fatores que tornam o uso das ferramentas de interpretação automatizada mais complexas [39]. Por outro lado, Wu e Zhang [73] comentam que tais fatores podem ocorrer por diversas razões, por exemplo, zonas de geologia complexa (por exemplo, perto de falhas), erros no processo de aquisição e baixa qualidade no modelo de velocidade de migração sísmica.

Portanto, as técnicas que abordam melhorias nos problemas aplicando aos processos de remoção de ruídos (*denoising*) [21], super-resolução [39], restauração (*deblurring*) [18], e interpolação [7, 47] estão em contínuo aprimoramento. Entretanto, tais técnicas, geralmente, se concentram na remoção de um tipo pré-determinado de ruído, como por exemplo ruídos incoerentes (branco, gaussiano, cintilação, etc.), ruídos coerentes (múltiplas, rotação do solo, onda de ar, etc.), na aumentação dos dados e na recuperação de traços sísmicos ausentes ou danificados. Em alguns casos, as técnicas tradicionais da sísmica de reflexão não são suficientes para obter a melhoria desejada.

Uma alternativa aos métodos tradicionais para potencializar e prover melhorias nas aplicações da sísmica de reflexão é baseada na técnica de Redes Neurais Profundas (*Deep Learning* [34]). Neste trabalho, propomos o desenvolvimento de Redes Neurais Profundas para remoção de ruídos e aumento de resolução através do ganho de altas frequências, sem perda das propriedades geológicas presentes nas imagens sísmicas. A atenuação de ruídos beneficiará o processo de interpretação sísmica, assim como, facilitará o uso de ferramentas automáticas. Já o ganho de altas frequências facilitará a identificação dos limites das reflexões, uma vez que o traço sísmico é uma superposição de muitas reflexões [33].

Nos dias atuais, a computação de alto desempenho permite que Redes Neurais Profundas sejam aplicadas em vários problemas da sísmica de exploração (por exemplo, [15, 39, 74, 82]). Mais especificamente, para a comunidade da sísmica de reflexão, o estado da arte está nas abordagens baseadas em Redes Adversárias Generativas (GANs, *Generative Adversarial Networks* [23]). As GANs consistem em dois modelos treinados como adversários: o gerador aprende a distribuição de dados reais e o discriminador aprende a distinguir amostras geradas (em outras palavras, sintéticas) de dados reais. As GANs são estado da arte em Visão Computacional para a geração de amostras sintéticas e tradução de imagem para imagem [49, 71].

Neste trabalho, exploramos arquiteturas de GANs que foram desenvolvidas para tradução de imagem para imagem. A ideia é traduzir uma imagem sísmica com ruído para uma outra imagem sísmica sem ruído, mantendo as propriedades principais dos dados (por exemplo, fase e amplitude).

No entanto, o treinamento das GANs envolve alguns desafios. Segundo Arjovsky e Bottou [36], além de instáveis, as GANs são difíceis de treinar, com inúmeras possibilidades de falhas durante o treinamento [4]. Outro desafio é a conversão da entrada e da saída das GANs (*input* e *output*) para dados sísmicos bruto ao invés de imagens, visto que a

conversão de dados sísmicos para imagens ocasionam em perda de precisão. Ainda nesse contexto, dados sísmicos são muito distintos em comparação com os conjuntos de dados de uso geral (para o qual as GANs utilizadas neste trabalho foi projetada), tornando a tarefa de geração ainda mais desafiadora.

Por fim, para treinar nosso modelo GAN é necessário um conjunto de dados sísmicos com ruídos e sem ruídos. Como obter dados reais sem ruídos é uma tarefa árdua (e aproximada, uma vez que pode não ser possível remover todo o ruído do dado real), também propomos a criação de uma base de dados sísmicos sintéticos. Para gerar esse tipo de dado, o desafio está na criação de estruturas geológicas e na modelagem dos ruídos.

1.1 Questões de Pesquisa

Para alcançarmos os objetivos deste trabalho, temos as seguintes questões de pesquisa (QP):

QP1: Para realizar a remoção de ruídos e o ganho de alta frequência, GANs projetadas para imagens são eficazes em dados sísmicos?

QP2: É possível utilizar GANs para a remoção de ruídos de dados sísmicos e o ganho de alta frequência?

QP3: Uma GAN treinada em dados sísmicos sintéticos (com e sem ruídos) é capaz de remover ruídos em dados sísmicos reais?

Em relação à QP2, o trabalho de Ganssle [21] levantou essa questão, usando a GAN Pix2Pix [29], para um escopo bastante restrito: remoção de ruído branco. Diante disso, consideramos que essa questão não foi respondida por Ganssle [21] e é uma questão de pesquisa relevante a ser respondida.

1.2 Contribuições

A principal contribuição deste trabalho é a proposição de abordagens baseadas em GANs para a remoção de ruídos e ganho de alta frequência, mantendo a integridade dos dados sísmicos. Destacamos que as nossas abordagens realizam o treinamento com dados sísmicos brutos, ou seja, não é necessário fazer a conversão dos dados sísmicos para imagens em tons de cinza (e nem tão pouco o contrário). Até onde sabemos, nossas abordagens baseadas em GANs são as primeiras a realizar tal treinamento na literatura.

Outra importante contribuição é o gerador de dados sísmicos sintéticos com e sem ruídos. Criamos duas versões do gerador de dados que utilizam funções desenvolvidas de acordo com as sugestões dos especialistas em sísmica para gerar estruturas geológicas.

Outras contribuições são as disponibilizações dos dados, dos modelos e dos códigos para a reprodução dos nossos resultados:

- Gerador de dados sísmicos sintéticos, disponível publicamente no GitHub (`https://github.com/jonlenes/seismic-synthetic-data-gen`).

- Disponibilizamos também as duas bases de dados sintéticas com 12.000 e 10.000 pares de dados sísmicos, criadas com as duas versões do gerador de dados.
- Modelos sísmicos treinados para remoção de ruídos. Os modelos pré-treinados também podem ser utilizados para transferência de aprendizado em outras tarefas sísmicas. Os modelos estão disponíveis publicamente no GitHub (`https://github.com/jonlenes/gan-seismic-denoising`).

1.3 Organização do Texto

O texto da dissertação está organizado da seguinte forma:

Capítulo 2 – Conceitos Relacionados: Apresentamos alguns conceitos necessários à compreensão deste trabalho de Mestrado, como as GANs, os seus principais problemas, desafios e as arquiteturas da Pix2Pix, Pix2PixHD e SPADE. Descrevemos também os dados sísmicos e o processo de experimentação sísmica realizado para a coleta e entendimento destes dados.

Capítulo 3 – Trabalhos Relacionados: Revisamos de forma sucinta a literatura dos últimos três anos referente à geração de dados sísmicos sintéticos e as aplicações de GANs na sísmica. Também apresentamos uma tabela resumo com todos os trabalhos, e destacamos aqueles que consideramos mais importantes, apresentando a relação entre esses trabalhos e o nosso.

Capítulo 4 – Metodologia Proposta: Descrevemos as etapas executadas para alcançar nossos objetivos de remoção de ruídos e ganho de alta frequência, mantendo a integridade dos dados sísmicos. Dentre eles o processo para a construção do nosso gerador de dados sísmicos sintéticos, adaptação das arquiteturas da Pix2Pix, Pix2PixHD e SPADE para o nosso problema, novas funções de custo e as métricas para validação dos nossos resultados.

Capítulo 5 – Experimentos e Resultados: Apresentamos os experimentos realizados e os resultados alcançados durante o desenvolvimento deste Mestrado, dentre eles, o processo experimental de extração das PSFs, os experimentos preliminares para verificar a viabilidade da pesquisa, e os experimentos relacionados ao gerador de dados sísmicos sintéticos. Com os dados sísmicos sintéticos construídos, apresentamos os experimentos com as GANs Pix2Pix, Pix2PixHD e SPADE, como avaliamos as diferentes funções de custo propostas e validamos os nossos modelos no Sigsbee (dado sintético) e em dados reais.

Capítulo 6 – Conclusões e Trabalhos Futuros: Avaliamos o trabalho criticamente, retomamos nossas questões de pesquisa, analisamos nossas descobertas e propomos orientações futuras para os problemas abordados.

Apêndice A – Fluxogramas: Apresentamos os fluxogramas de desenvolvimento do gerador de dados, treinamento dos modelos, avaliação dos resultados e utilização dos modelos para predizer imagens sísmicas.

Capítulo 2

Conceitos Relacionados

Neste capítulo, apresentamos alguns conceitos necessários à compreensão deste projeto de pesquisa. Na Seção 2.1, apresentamos as redes adversárias generativas, os seus principais problemas, desafios e as arquiteturas propostas na literatura. Na Seção 2.2, descrevemos os dados sísmicos e o processo de experimentação sísmica realizado para a coleta e entendimento destes dados.

2.1 Redes Adversárias Generativas (GANs)

As Redes Adversárias Generativas (*Generative Adversarial Networks*, ou GANs) foram propostas por Goodfellow et al. [23] em 2014 como sendo um método capaz de produzir amostras sintéticas realistas, ou seja, semelhantes a um conjunto de dados reais. Dada a capacidade das GANs de aprender uma distribuição e criar uma nova amostra, elas têm sido usadas para gerar diversos tipos de dados, como imagens, vídeos e músicas [4]. Na Figura 2.1, são apresentadas algumas imagens geradas pela GAN SPADE [49].

Figura 2.1: Imagens sintéticas geradas pela SPADE. Figura adaptada de Park et al. [49].

Tipicamente, as arquiteturas das GANs são compostas por duas redes neurais profundas. O gerador $G_\Theta(Z) = G(Z;\Theta)$ tenta aprender através de um mapeamento do ruído de entrada para uma amostra realista, com $G_\Theta : \mathbb{Z} \to \mathbb{X}$ onde $\mathbb{Z}$ é um espaço vetores de

tamanho n_Z ou uma imagem (com até três canais de cores) $n_Z \times m_Z \times c_Z$ e $\mathbb{X}$ é espaço das imagens (com três canais de cores) $n_X \times m_X \times 3$. Mais ainda, Θ representa um vetor dos parâmetros da rede de G. Já o discriminador $D_\Lambda(X) = D(X;\Lambda)$ recebe os dados reais do conjunto de treinamento ou os dados falsos do gerador, e com isso ele classifica a probabilidade de cada uma das entradas no intervalo fechado entre zero (falsa) e um (real) [4], sendo $D_\Lambda : \mathbb{X} \to [0,1]$ e Λ representa um vetor dos parâmetros da rede de D. Para simplificar a notação usaremos apenas D e G. Ademais, quando nos referimos à procura dos melhores D e G, estamos buscando os melhores parâmetros Λ e Θ, respectivamente. Retomando a explicação, o gerador recebe o ruído e tenta gerar amostras que possam enganar o discriminador (ou seja, tenta aprender a produzir novos dados com as mesmas estatísticas que o conjunto de treinamento). O modelo generativo pode ser considerado análogo a um falsificador de moeda que tenta não ser descoberto, enquanto o modelo discriminativo é análogo à polícia que tenta identificar se a moeda é falsa.

As GANs baseiam-se no jogo não-cooperativo de soma zero (ou teorema *minmax* [23]), no qual um oponente quer maximizar suas próprias ações, e o outro quer minimizá-las. Neste caso, as redes G e D competem em um jogo de dois jogadores sua distância medida pelo funcional $\mathcal{F}_{\text{GAN}}(G,D)$ definido por

$$\mathcal{F}_{\text{GAN}}(D,G) = E_{(X\sim p_{\text{data}})}\left[\log D(X)\right] + E_{(Z\sim p_{\text{noise}})}\left[\log(1 - D(G(Z)))\right], \tag{2.1}$$

onde X é um dado amostral, $p_{\text{data}} \subset \mathbb{X}$ é a distribuição dos dados amostrais reais, Z é um ruído e $p_{\text{noise}} \subset \mathbb{Z}$ é a distribuição do ruído. Além disso, durante o treinamento, o gerador se torna progressivamente melhor na criação de dados similares aos reais, enquanto o discriminador se torna melhor em diferenciá-las (veja Figura 2.2). Portanto, o processo atinge o equilíbrio quando o discriminador não consegue mais distinguir imagens reais de falsificações [23]. Quando treinados corretamente, é esperado que tanto o gerador quanto o discriminador atinjam o equilíbrio de *Nash*, onde não é possível um encontrar vantagens sobre o outro e o jogador não muda sua ação independentemente da ação do oponente [4, 42].

Inicialmente, os modelos gerador e discriminador eram implementados como *Multilayer Perceptrons* (MLP)[1]. Radford et al. [53] propuseram em 2015 a *Deep Convolutional* GAN (DCGAN), onde os modelos foram implementados como redes neurais convolucionais profundas. Diversas arquiteturas foram propostas desde então, mas a maioria segue a mesma proposta da DCGAN.

A arquitetura geral da DCGAN, e das demais GANs estudadas neste trabalho, é apresentada na Figura 2.3, onde *generator* (gerador) e *discriminator* (discriminador) representam redes neurais profundas, e *training set* é o conjunto de dados com amostras reais. O gerador recebe um vetor de ruído aleatório (*random noise*) e gera uma saída (*fake image*) que é passada para o discriminador, que por sua vez determina se a imagem é provinda da distribuição do ruído ou dos dados de treinamento (*real/fake*).

Mais especificamente, o ruído é repassado para a primeira camada do gerador, cha-

[1]MLP é um algoritmo de aprendizado supervisionado que aprende uma função $F(\cdot) : R^m \to R^n$ treinando em um conjunto de dados, onde m é o número de dimensões para entrada e n é o número de dimensões para saída [58].

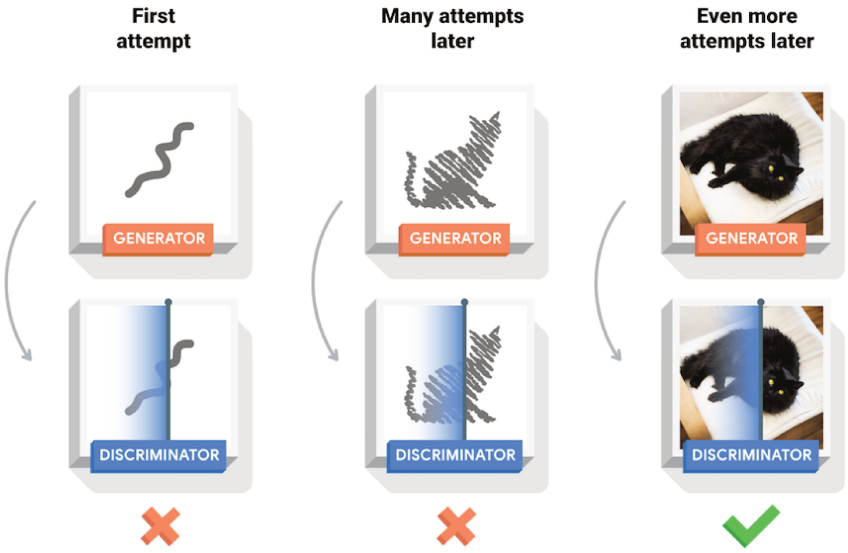


Figura 2.2: Exemplo do processo de treinamento de uma GAN: na primeira tentativa (*first attempt*) o gerador ainda não tem nenhuma informação para gerar um gato, logo gera qualquer imagem, que é repassada para o discriminador. A imagem é facilmente identificada como falsa pelo discriminador, que devolve o *feedback* para o gerador. Esse processo continua até que muitas tentativas depois (*many attempts later*) o gerador já aprendeu a distribuição geral do gato no que deve ser gerado, mas ainda pode pode ser distinguido do gato real pelo discriminador. Quando o gerador é capaz de gerar uma imagem que não possa ser distinguida da imagem real (*even more attempts later*), o treinamento está em equilíbrio. Figura reproduzida de Abadi et al. [1].

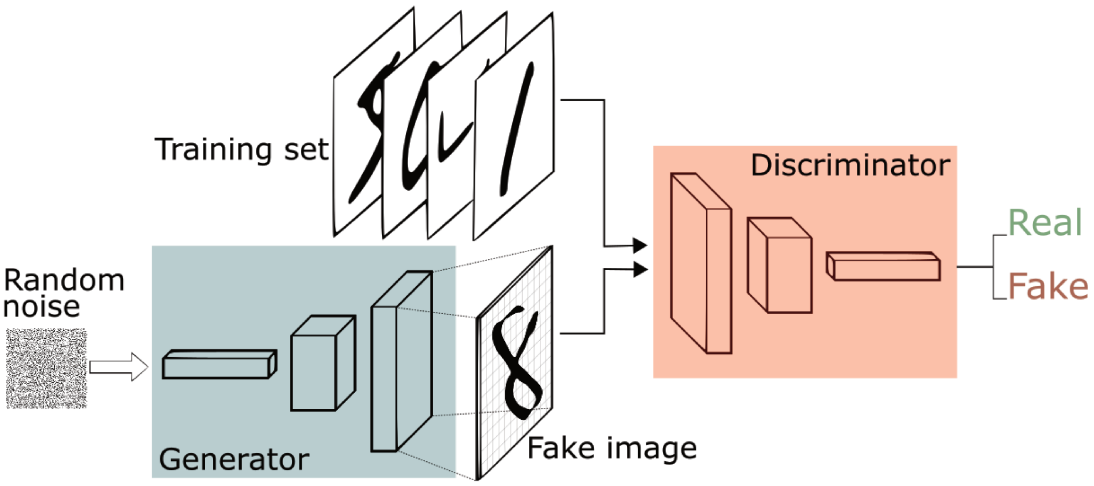


Figura 2.3: Arquitetura geral das GANs. Figura reproduzida de Silva [60].

mada de *project and reshape*, que expande os dados de entrada. Em seguida temos os blocos convolucionais que duplicam o tamanho do primeiro e segundo eixo, e diminui gradualmente o número de canais até a quantidade desejada de canais na imagem de saída. Essas camadas convolucionais são usadas para realizar *up-sampling* (veja Figura 2.4). Já

a arquitetura do discriminador é semelhante às redes de classificação e, que para este caso, realiza as operações opostas ao gerador, tendo o resultado resumido a uma classificação binária: real ou falsa (veja Figura 2.5).

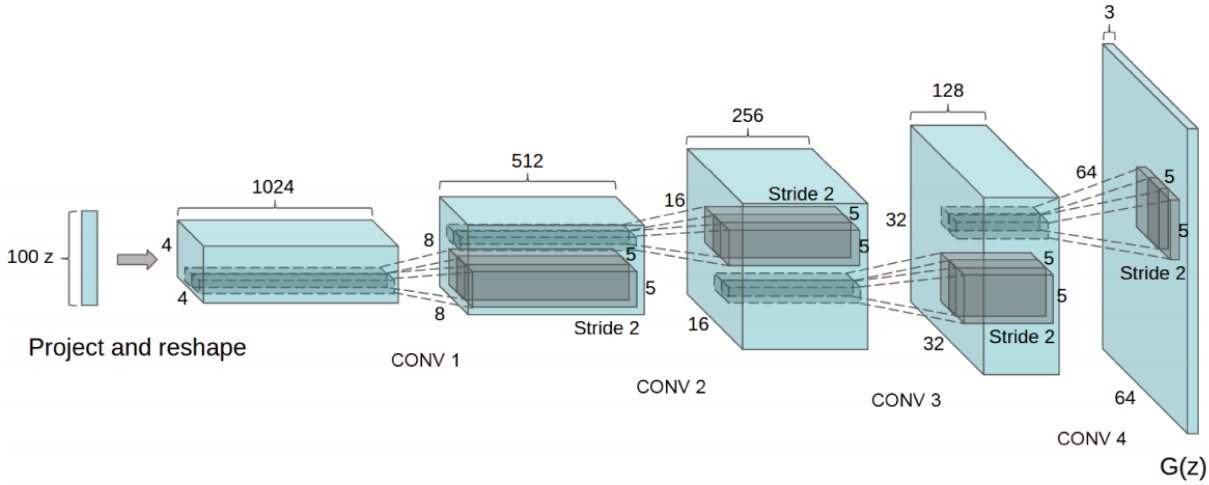


Figura 2.4: Arquitetura do gerador da DCGAN: uma distribuição uniforme de 100 dimensões Z é projetada em uma pequena representação espacial convolucional com muitos mapas de características (1024). Em seguida, são aplicadas quatro convoluções e o resultado é convertido para uma representação de alto nível em uma imagem de 64×64 píxeis. Figura reproduzida de Radford et al. [53].

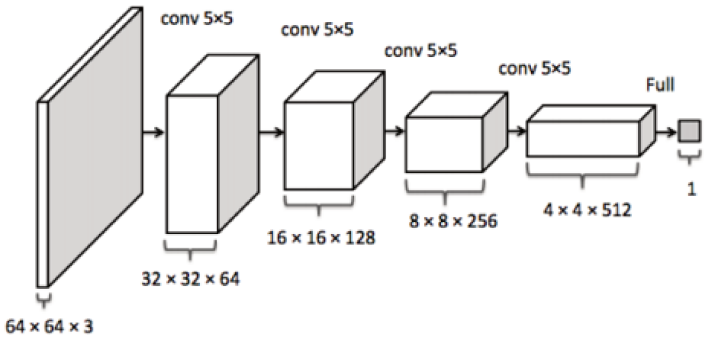


Figura 2.5: Arquitetura do discriminador da DCGAN: a entrada da rede é uma imagem e a saída é um único valor indicando a probabilidade da imagem ser falsa. Figura adaptada de Zhang [84].

2.1.1 Problemas

As GANs são conhecidas por sua instabilidade para treinar, geralmente resultando em geradores que produzem saídas sem sentido [53]. Segundo Arjovsky e Bottou [36], além de instáveis, são difíceis de treinar, com inúmeras possibilidades de falhas durante o treinamento [4]. Melhorar essas características é importante para evitar problemas com o modelo e seus resultados. Nesta seção, apresentaremos alguns destes problemas.

O primeiro deles refere-se justamente à estabilidade do treinamento, pois uma GAN exige um cuidadoso equilíbrio entre o discriminador e o gerador. Apesar de estarem lutando entre si, eles ainda precisam um do outro para melhorar [4]. Por exemplo, o discriminador pode sobrecarregar o gerador e tornar-se absolutamente certo de que os dados gerados são falsos. Consequentemente, a função de custo atinge zero e não há gradiente disponível para ser enviado ao gerador. Para tal, é recomendado por Bernico [4] o ajuste das arquiteturas e hiperparâmetros a fim de evitar que um domine o outro.

Já o *mode collapse*, problema de difícil resolução em GANs [3, 36], caracteriza-se como sendo a produção de amostras de baixa variedade pelo gerador [57]. Nesse caso, o gerador explora um único aspecto da distribuição de treinamento, de modo a enganar o discriminador produzindo imagens similares às originais. Um exemplo deste problema pode ser visto na Figura 2.6.

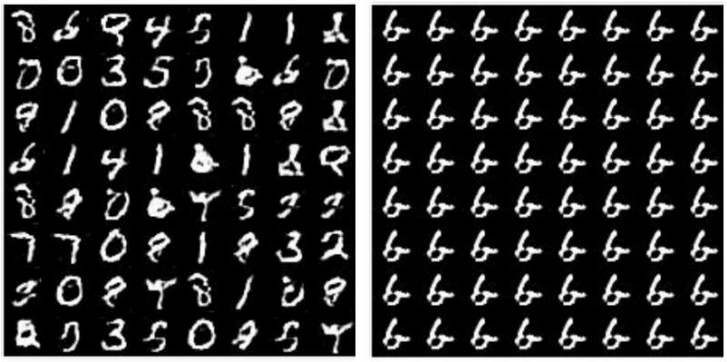


Figura 2.6: À esquerda, o modelo produz todas as 10 classes da base de dados MNIST, enquanto à direita é criada apenas uma classe. Figura adaptada de Metz et al. [43].

A seleção de hiperparâmetros também é um desafio, pois em redes neurais a escolha destes é frequentemente auxiliada pelos valores da função de custo que se deseja minimizar. Entretanto, as GANs utilizam-se de uma perda relativa entre adversários, implicando que a minimização das funções de custo de G e D pode não ser a solução ótima para o problema, dificultando a escolha dos hiperparâmetros mais adequados.

Além destes, existem outros problemas que podem afetar o treinamento das GANs. Para mais informações, recomendamos a leitura do trabalho de Arjovsky e Bottou [36] e Bissoto et al. [6]. Na seção seguinte, apresentaremos algumas arquiteturas propostas que visam diminuir os problemas das GANs e que estão diretamente relacionadas com as nossas abordagens propostas.

2.1.2 Arquiteturas

Mais de 500 arquiteturas de GANs[2] foram propostas visando à diminuição de seus problemas e da ampliação da sua aplicabilidade. Nesta seção, apresentaremos as que consideramos mais relevantes ao nosso trabalho.

[2]Lista das arquiteturas de GANs: `https://github.com/hindupuravinash/the-gan-zoo`.

Pix2Pix

Isola et al. [29] introduziram em 2017 uma GAN que adiciona uma imagem como entrada condicional ao gerador. Em contraste com a proposta original das GANs ($G : \mathbb{Z} \rightarrow \mathbb{X}$ e $D : \mathbb{X} \rightarrow [0, 1]$), as GANs condicionais aprendem um mapeamento da imagem observada e do ruído aleatório, ou seja, $G : \mathbb{X} \times \mathbb{Z} \rightarrow \mathbb{X}$ e $D : \mathbb{X} \times \mathbb{X} \rightarrow [0, 1]$. Tal rede, chamada Pix2Pix, foi projetada para executar tradução de imagem em imagem, onde é utilizada uma GAN condicional [44] para aprender uma função que mapeia uma imagem de entrada para uma imagem de saída (veja Figura 2.7 para exemplos de tradução de imagem para imagem realizados pela Pix2Pix). A Pix2Pix segue a mesma proposta de Goodfellow et al. [23], em que a rede é composta pelo gerador e discriminador: o gerador transforma a imagem de entrada para obter a imagem de saída; o discriminador analisa o par de entrada-alvo e o par de entrada-saída mensura a similaridade entre elas e computa a probabilidade delas terem sido produzidas pelo gerador.

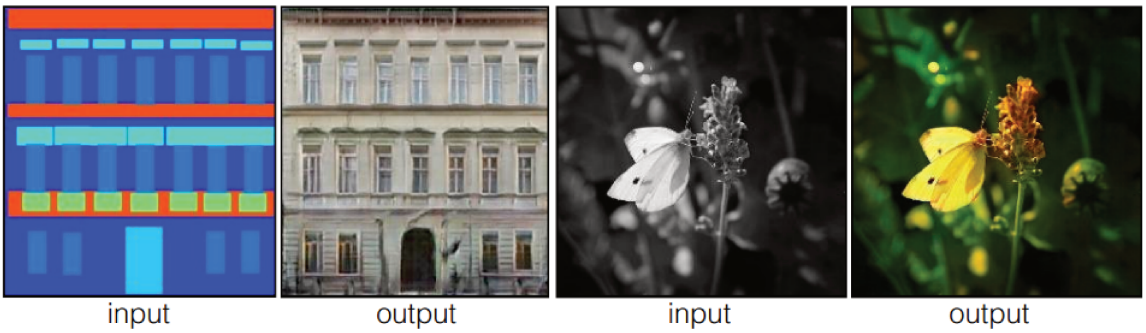


Figura 2.7: Tradução de imagem para imagem, onde a imagem de entrada (*input*) é traduzida para uma de saída (*output*). Figura adaptada de Isola et al. [29].

Uma das mudanças introduzidas pela Pix2Pix foi a utilização de uma perda de conteúdo $\mathcal{F}_{\text{lost}}(G)$ entre a imagem de saída da rede e a imagem real [17], isto é,

$$\mathcal{F}_{\text{lost}}(G) = E_{(X,Y\sim p_{\text{data}};Z\sim p_{\text{noise}})}\left[||Y - G(X, Z)||_1\right] , \tag{2.2}$$

onde $||.||_1$ é a norma L_1 e pode ser definida como a soma das magnitudes das matrizes [68]. A Equação 2.2 é ponderada por um fator $\lambda > 0$, gerando uma função de perda final para ser otimizada descrita pela Equação 2.3:

$$\mathcal{L}_{\text{cGAN}}(G_*, D_*) = \min_G \max_D \left[\mathcal{F}_{\text{cGAN}}(G, D) + \lambda\mathcal{F}_{\text{lost}}(G)\right] , \tag{2.3}$$

onde $\mathcal{F}_{\text{cGAN}}(G, D)$ é a função de custo da cGAN proposta por Mirza e Osindero [44] e é definida conforme a seguir,

$$\begin{aligned} \mathcal{F}_{\text{cGAN}}(G, D) &= E_{(X,Y\sim p_{\text{data}})}\left[\log D(X, Y)\right] \\ &+ E_{(X\sim p_{\text{data}};Z\sim p_{\text{noise}})}\left[\log(1 - D(X, G(X, Z)))\right] . \end{aligned} \tag{2.4}$$

Com essa nova função de custo, a Pix2Pix segue a mesma metodologia de treinamento

proposta por Goodfellow et al. [23], conforme apresentado na Figura 2.8.

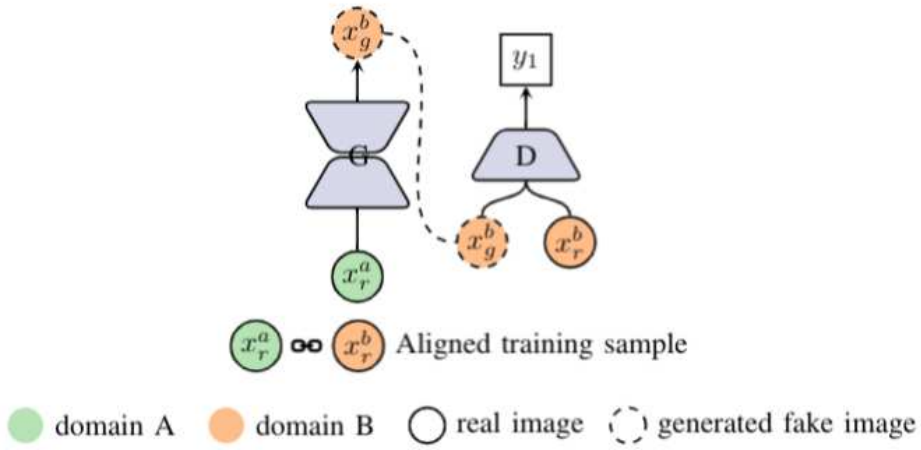


Figura 2.8: Processo de treinamento da Pix2Pix: primeiro G recebe uma imagem real do domínio A (x_r^a), gerando uma imagem falsa do domínio B (x_g^b). Em seguida, a imagem falsa é passada para o discriminador juntamente com a imagem real do domínio B (x_r^b), que analisa o par entrada-alvo e realiza a classificação em real e falsa. Figura reproduzida de Yi et al. [77].

Adicionalmente, a Pix2Pix propõe modificações na arquitetura nos modelos de G e D que aprimoram diversos aspectos dos problemas das GANs apresentada na seção anterior:

- G recebe apenas uma imagem como entrada, que, diferentemente das GANs tradicionais, não tem ruído do espaço latente. Ao invés disto, o ruído provém das camadas de *dropout*[3] aplicadas em várias camadas do gerador, tanto no treinamento quanto no teste.

- O gerador utiliza a arquitetura U-Net proposta por Ronneberger et al. [56] em que a imagem de entrada tem suas camadas reduzidas até atingir uma camada de gargalo, onde a representação é aumentada novamente em algumas camadas até a imagem final.

- Adição da *skip connections*[4] à U-Net entre as camadas correspondentes de *down-sample* e *up-sample* para melhorar o compartilhamento de informações entre entrada e saída (veja a Figura 2.9a).

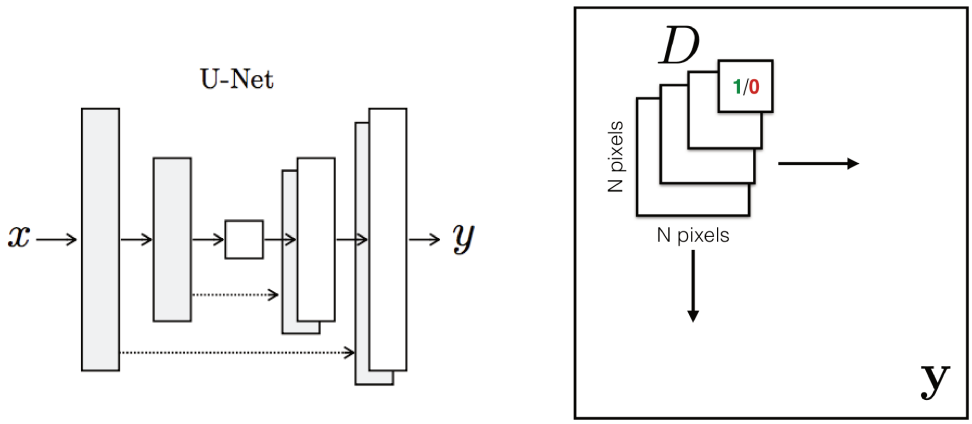
- D recebe duas imagens: uma imagem do domínio de origem e uma imagem do domínio de destino e prediz a probabilidade da imagem do domínio de destino ser real ou gerada.

- O discriminador usa um PatchGAN (semelhante ao proposto por Li e Wand [35]), uma *Deep Convolutional Neural Network* (DCNN) projetada para classificar *patches*

[3] *Dropout* consiste em descartar aleatoriamente neurônios e suas conexões da rede neural durante o treinamento [64].

[4] *Skip connections* são conexões extras entre nós em diferentes camadas de uma rede neural que ignoram uma ou mais camadas do processamento não linear [48].

de uma imagem de entrada como reais ou falsas, ao invés da imagem inteira. A saída da rede é um *feature map* de previsões reais/falsas, onde se pode calcular uma média para obter o resultado final (veja a Figura 2.9b).



(a) (b)

Figura 2.9: (a) U-Net com *skip connections* entre camadas espelhadas na pilha de camadas de *down-sample* e *up-sample*, onde x representa a imagem de entrada e y a imagem de saída. (b) D penaliza se cada *patch* sobreposto na saída parecer falso. Figura reproduzida de Isola et al. [29].

Pix2PixHD

Wang et al. [71] apresentam a rede Pix2PixHD como um novo método para sintetizar imagens de alta resolução a partir de mapas semânticos[5] utilizando também a ideia da cGANs. Essa rede é capaz de gerar imagens até a resolução de 2048×1024. Algumas das principais melhorias em relação à Pix2Pix são:

- Gerador *coarse-to-fine*: em que o gerador é composto por duas redes, G_1 e G_2, denominados como gerador global e rede local de aprimoramento, respectivamente. O Gerador G é composto pela tupla $\{G_1, G_2\}$ conforme apresentado na Figura 2.10. O gerador global G_1 (baseado na arquitetura proposta por Johnson et al. [31]) é treinado em imagens de baixa resolução, enquanto o gerador local G_2 é treinado recebendo uma imagem com a resolução normal e as *features* geradas por G_1. Aqui, novamente, temos $G : \mathbb{X} \rightarrow \mathbb{X}$. A combinação desses dois geradores, trabalhando em diferentes resoluções apresentou melhorias para diversas tarefas com imagens de alta resolução, conforme apresentado por Wang et al. [71].

- Discriminadores *multi-scale*: três discriminadores que possuem a mesma estrutura de rede, mas operam em diferentes escalas de imagem, foram usados para lidar com a necessidade de grandes campos receptivos.

[5]Mapa Semântico consiste em mapear uma imagem que foi rotulada em relação a cada um de seus pixeis de acordo com a classe correspondente ao que está sendo representado.

- Juntamente com a nova arquitetura do discriminador, os autores adicionaram a função de custo, chamada *feature matching* (FM), no discriminador e uma outra, chamada *perceptual* [31], no gerador. A função FM estabiliza o treinamento, pois o gerador precisa produzir estatísticas naturais em várias escalas, além de forçar o gerador para minimizar a diferença estatística entre os recursos das imagens reais e as imagens geradas [27]. Especificamente, são extraídas *features* de várias camadas do discriminador, e aprende se a combinar essas representações intermediárias da imagem real e da imagem sintetizada. Para facilitar a apresentação da função de custo foi denotado o extrator de *features* de camada i do discriminador D_k como $D_k^{(i)}$ com $D_k : \mathbb{X} \to [0, 1]$. Além do mais, tal função de perda pode ser descrita como:

$$\mathcal{F}_{\mathrm{FM}}(G, D_k) = E_{(X\sim p_{\mathrm{data}};Z\sim p_{\mathrm{noise}})}\left[\sum_{i=1}^{N_c}\frac{1}{N_i}\left\|D_k^{(i)}(Z, X) - D_k^{(i)}(Z, G(Z))\right\|_1\right], \quad (2.5)$$

onde N_c é o número total de camadas e N_i denota o número de elementos em cada camada. Por outro lado, a função *perceptual* mostrou-se bastante eficaz para os experimentos realizados de ajustes finos na geração de imagem [71], podendo ser definida como:

$$\mathcal{F}_{\mathrm{P}}(G) = E_{(X\sim p_{\mathrm{data}};Z\sim p_{\mathrm{noise}})}\left[\sum_{i=1}^{N}\frac{1}{M_i}\left\|F^{(i)}(X) - F^{(i)}(G(Z))\right\|_1\right], \quad (2.6)$$

onde $F^{(i)}$ denota a camada i com M_i elementos da rede VGG (mais informações podem ser encontradas em Simonyan e Zisserman [61]).

O objetivo completo combina a função de custo da GAN, a *feature matching* e a *perceptual* do seguinte modo:

$$\begin{aligned}\mathcal{L}_{\mathrm{HD}}(G_*, \{D_k\}_*) &= \min_G\left(\left(\max_{D_1,D_2,D_3}\sum_{k=1,2,3}\mathcal{F}_{\mathrm{HD}}\left(G, D_k\right)\right)\right. \\ &+ \left.\lambda_1\sum_{k=1,2,3}\mathcal{F}_{\mathrm{FM}}\left(G, D_k\right) + \lambda_2\mathcal{F}_{\mathrm{P}}(G)\right), \quad (2.7)\end{aligned}$$

onde λ_1 e λ_2 são os pesos de cada uma das funções na função de custo total e $\mathcal{F}_{\mathrm{HD}}(G, D_k)$ é a função de custo desta GAN e é definida como,

$$\begin{aligned}\mathcal{F}_{\mathrm{HD}}(G, D_k) &= E_{(X\sim p_{\mathrm{data}};Z\sim p_{\mathrm{noise}})}\left[\log D_k(Z, X)\right] \\ &+ E_{(X\sim p_{\mathrm{data}};Z\sim p_{\mathrm{noise}})}\left[\log(1 - D_k(Z, G(Z)))\right]. \quad (2.8)\end{aligned}$$

As melhorias na Pix2Pix apresentadas pela Pix2PixHD têm um impacto significativo sobre as imagens de alta resolução, mas essas melhorias também podem ajudar na tarefa de tradução de imagem para imagem em baixa resolução. A Figura 2.11 mostra alguns exemplos de tradução de mapas semânticos para imagem.

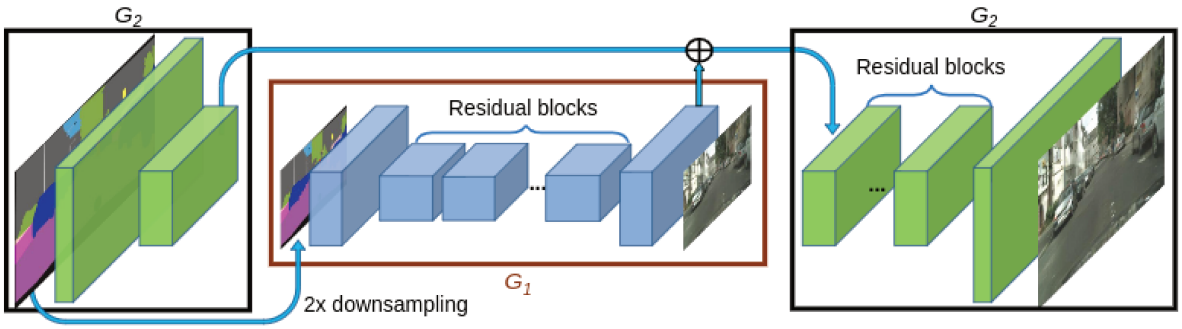


Figura 2.10: Arquitetura do gerador da Pix2PixHD: primeiro é treinada uma rede residual G_1 em imagens de baixa resolução; em seguida, outra rede residual G_2 é anexada a G_1 e as duas redes são treinadas em conjunto em imagens de alta resolução. Especificamente, a entrada para os blocos residuais em G_2 é a soma dos elementos do mapa de *features* de G_2 e o último mapa de *features* de G_1 [71].

Figura 2.11: Tradução de mapas semânticos para imagem: em cada imagem é apresentado o mapa semântico à esquerda e a imagem gerada à direita. Figura adaptada de Wang et al. [71].

SPADE

Park et al. [49] apresentam uma normalização espacialmente adaptativa[6] através de uma nova camada para sintetizar imagens foto-realistas, denominada SPADE. Os métodos anteriores (como a Pix2PixHD, por exemplo) repassa o mapa semântico diretamente como entrada da rede profunda, que é então processado através de pilhas de camadas de convolução, normalização e camadas não lineares. Enquanto isso, a proposta da SPADE é que o mapa semântico de entrada, que devem ser valores inteiros, sejam repassados para a camada de normalização.

A Figura 2.12 apresenta o projeto da normalização SPADE, onde α e β são os parâmetros de modulação aprendidos. Na prática, esses parâmetros de normalização servem para que se possa aprender a melhor normalização possível para cada um dos *labels* presentes no mapa de segmentação, ao invés de utilizar a mesma normalização para todos os *labels*. Com isso, não é necessário passar o mapa de segmentação para a primeira camada do gerador, pois esses parâmetros de modulação aprendidos codificaram informações suficientes sobre o mapa semântico, e o gerador pode utilizar um vetor aleatório como entrada.

A Figura 2.13 ilustra a arquitetura do gerador SPADE, que emprega vários blocos residuais da ResNet [25] com camadas de *up-sampling*, onde todos os parâmetros de modulação de todas as camadas de normalização são treinados. Como cada bloco residual

[6]Adaptativa porque é uma camada de normalização com parâmetros treináveis.

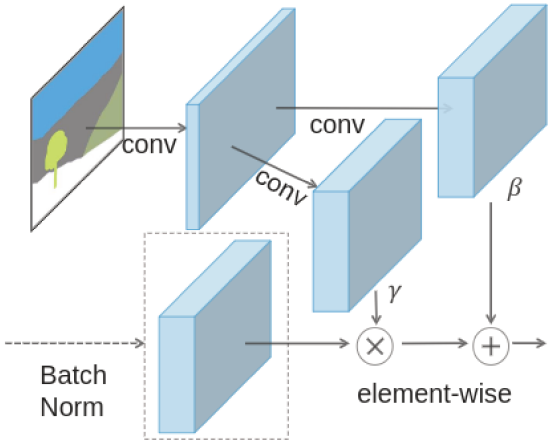


Figura 2.12: Normalização SPADE: o mapa semântico é primeiro projetado em um espaço de *embedding* e, em seguida, é aplicada uma convolução para produzir os parâmetros de modulação α e β, que são multiplicados e adicionados ao mapa de ativação normalizado *element-wise*. Figura reproduzida de Park et al. [49].

opera em uma escala diferente, foi reduzido o tamanho do mapa semântico inicial para corresponder as suas respectivas resoluções de cada bloco. Quanto ao discriminador e as perdas, foram utilizados os mesmos da Pix2PixHD [71].

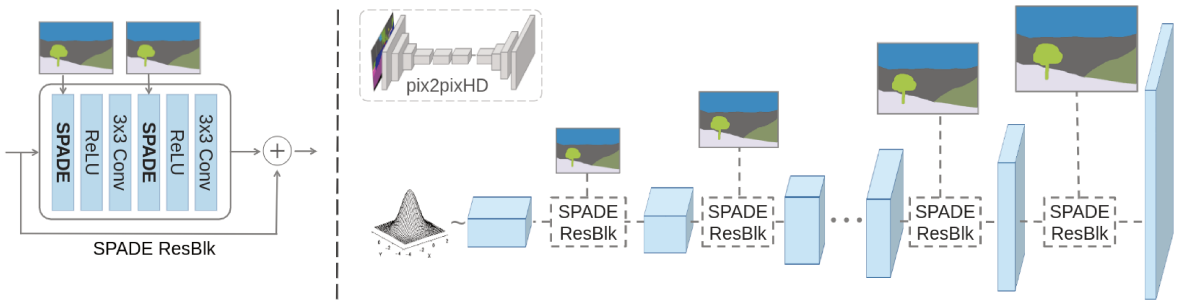


Figura 2.13: No gerador SPADE, cada camada de normalização utiliza um mapa semântico com a respectiva resolução do mapa de ativação da camada anterior. À esquerda é apresentado um bloco residual com a SPADE, enquanto à direita é apresentado o gerador que contém uma série de blocos residuais SPADE com camadas de *up-sampling*. Figura reproduzida de Park et al. [49].

Os resultados dos experimentos com a SPADE em vários conjuntos de dados demonstram a vantagem do método proposto sobre as abordagens existentes, tanto em relação à fidelidade visual quanto ao alinhamento com os mapas de entrada. A Figura 2.14 apresenta alguns exemplos de tradução de mapas semânticos para imagens (para mais detalhes e exemplos consulte Park et al. [49]).

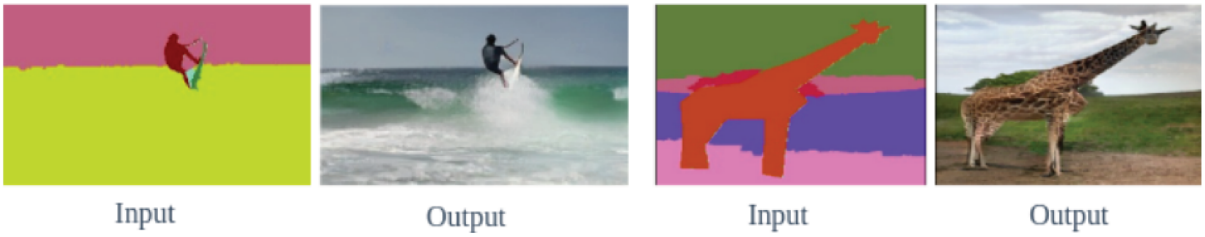


Figura 2.14: Tradução de mapas semânticos para imagem usando a SPADE: em cada imagem é apresentado o mapa semântico (*input*) e a imagem gerada (*output*). Figura adaptada de Park et al. [49].

2.2 Imagens Sísmicas

O *levantamento sísmico* é um dos componentes essenciais para a exploração de petróleo e gás. Em resumo, os dados coletados através deste processo permitem aos geocientistas visualizarem as camadas subterrâneas da Terra usando ondas sísmicas produzidas de forma artificial, geralmente, na superfície da Terra, com frequências variando entre 5 Hz até 100 Hz, que penetram em seu interior e "mapeiam" as subestruturas geológicas. Estes dados ajudam a determinar a localização dos reservatórios e selecionar os melhores locais para perfuração [69].

Para o levantamento sísmico são utilizadas fontes sísmicas com intensidade controlada que estimulam ondas sísmicas no interior da Terra. Tais fontes podem ser de vários tipos, e.g, cargas controladas de dinamite, caminhões vibradores, ou por canhões de ar em ambiente marinho. Um dos fatores que influenciam em qual método será utilizado é o local onde será realizado os experimentos: na terra ou no mar. Quando o levantamento ocorre em terra, as ondas sísmicas vibram, tal vibração propaga no subsolo e quando encontra diferenças entre os materiais que compõem a subsuperfície, tais ondas refletem e refratam. As ondas refletidas retornam à superfície sendo capturas por um dispositivo de gravação chamado geofone que estão especificamente posicionados no terreno. Por outro lado, quando o levantamento é realizado no mar, um dispositivo chamado hidrofone é usado para capturar as ondas sonoras [28, 69]. Esses dispositivos de gravação utilizados são chamados de receptor (um exemplo de levantamento sísmico é apresentado na Figura 2.15).

Cada tipo de fonte gera um pulso característico, que resulta em dados com assinaturas diferentes [28]. O caminho de cada onda individual revela um contorno específico da estrutura geológica. Cada receptor realiza o registro temporal da pressão ou movimento das partículas ao seu redor, tendo uma determinada taxa de amostragem, sendo comumente entre 1 ms até 8 ms para a sísmica de reflexão [5].

Em superfície, o registro realizado pelos receptores em função do tempo do sinal que foi medido é chamado de *traço sísmico*. Este é uma combinação da resposta do ambiente com o pulso de origem utilizado e está, sempre, associado a um par fonte× receptor; para uma fonte e um receptor, há apenas um traço que representa as amplitudes recebidas pelo receptor ao longo do tempo provindo da fonte associada [5, 28, 78].

Após este processo, esses dados brutos coletados são armazenados em arquivos digitais, por exemplo, *Society of Exploration Geophysicists format* (SEG-Y) e *Seismic Unix*

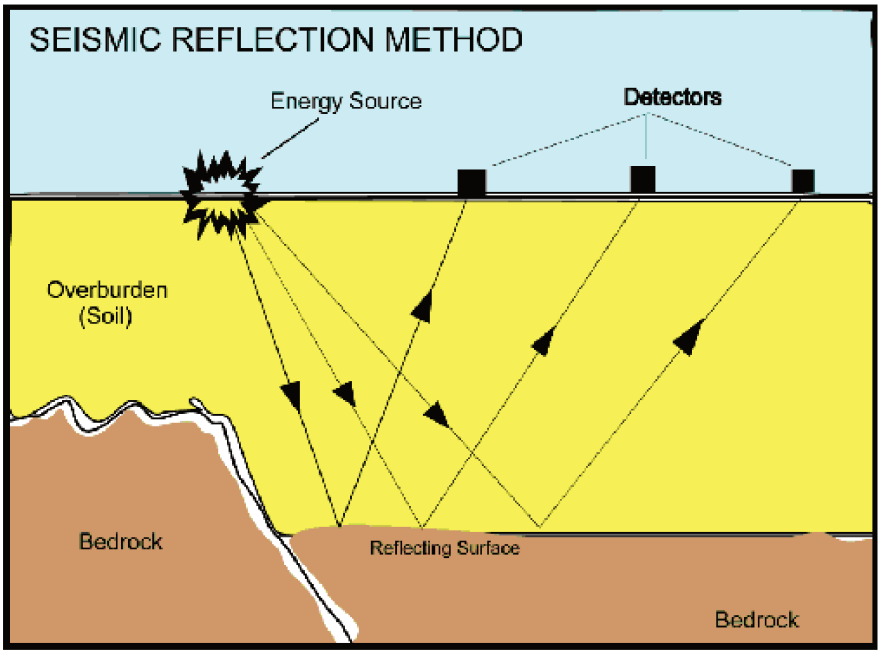


Figura 2.15: O tiro disparado pela fonte (*Energy Source*) causa a propagação das ondas até uma subsuperfície refletora (*Reflecting Surface*) que são refletidas e capturadas pelos receptores (*Detectors*). Figura reproduzida de Resources [54].

(SU), e devem passar por uma série de processamentos para que sejam representativos da subsuperfície. Em particular, o processo de migração (ou imageamento sísmico) reposiciona os eventos de reflexão para suas posições corretas em relação ao tempo ou à profundidade. O dado migrado pode então ser chamado de uma imagem sísmica, pois somente após a migração podemos dizer que há correlação entre a disposição dos eventos no dado e a disposição das estruturas geológicas [78].

Neste trabalho, utilizaremos os dados migrados em tempo (que são nossas imagens sísmicas), conforme exemplo apresentado na Figura 2.16. Resumidamente, com os dados migrados, a etapa de interpretação pode ser realizada com mais eficiência e incertezas podem ser minimizadas. Após tal processo, os especialistas analisam as imagens sísmicas para descrever o conteúdo da subsuperfície. Uma melhoria nestes dados pode representar uma diminuição significativa no tempo gasto por estes especialistas, sendo este um dos objetivos deste trabalho.

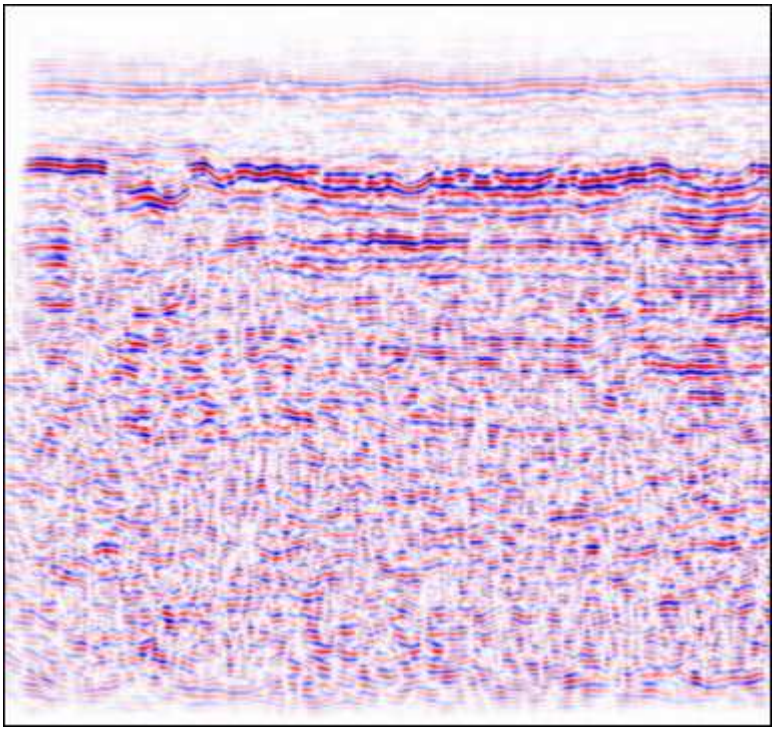


Figura 2.16: Imagem sísmica do dados da bacia brasileira do Solimões. O eixo vertical representa o tempo vertical, o eixo horizontal representa o deslocamento lateral e as cores representam os valores das amplitudes, variando de -1 (azul) a $+1$ (vermelho).

Capítulo 3

Trabalhos Relacionados

O advento de novos paradigmas *Deep Learning* (e *Machine Learning*) está permitindo o desenvolvimento de novas soluções para enfrentar os desafios impostos pelas aplicações nas imagens sísmicas. Especificamente, as redes neurais convolucionais (*convolutional neural networks*, CNNs) têm sido investigadas como novas ferramentas para o processamento de imagens sísmicas [51]. Interpolação, reconstrução, remoção de ruídos (*denoising*), ganho de frequência e modelo de velocidade são algumas das principais áreas que estão sendo exploradas com técnicas de *Deep Learning*.

Apesar da vasta literatura de *Deep Learning* referente aos dados sísmicos, neste capítulo apresentamos trabalhos recentes que consideramos de maior relevância, e preferencialmente aqueles que utilizam GANs. Apresentamos um sucinto estudo sobre como as CNNs foram utilizadas, suas vantagens e como se relacionam ao nosso trabalho. De modo geral, os trabalhos apresentados neste capítulo visam o aprimoramento dos dados sísmicos ou de seu processo para prover dados mais precisos e processos mais acurados.

Esse capítulo está organizado em três partes. Na Seção 3.1, apresentamos a literatura referente à geração de dados sísmicos sintéticos. Na Seção 3.2, apresentamos os trabalhos de aplicações de GANs na Sísmica. Para sumarizar as informações apresentadas, na Seção 3.3, apresentamos uma tabela que resume os trabalhos deste capítulo, e destacamos aqueles que consideramos mais importantes, discutindo a relação com o nosso trabalho.

3.1 Geração de Dados Sísmicos Sintéticos

Nos últimos anos, a indústria de petróleo e gás aumentou o interesse em aplicar técnicas de *Machine Learning* para acelerar o processo de interpretação sísmica, considerada uma tarefa demorada e centrada no ser humano. Embora técnicas de *Machine Learning* tenham sido utilizada com sucesso em muitas aplicações, desde a segmentação estratigráfica até a detecção de domos de sal, um gargalo comum é a necessidade de grandes quantidades de dados anotados e de alta qualidade [19].

Para esse problema, alguns trabalhos [45, 50] utilizam/modificam dados sintéticos disponíveis publicamente (por exemplo, o dado disponibilizado por Martin [41]), e outros trabalhos [14, 19, 45, 50, 62, 74], e também este trabalho, propõem a geração de dados sísmicos sintéticos, visto que os existentes não atendem às necessidades do problema em

questão. Alguns deles baseiam-se em métodos tradicionais [62], enquanto outros baseiam-se em técnicas de *Deep Learning* [19]. Ferreira et al. [19] apresentam uma investigação sobre a geração de imagens sísmicas sintéticas com base em esboços (*sketches*) usando GANs (veja a Figura 3.1). Seus resultados indicam que imagens sísmicas realistas podem ser sintetizadas usando esboços simples. Entretanto, essa abordagem necessita dos esboços, sendo este o desafio para a geração de dados nesse contexto.

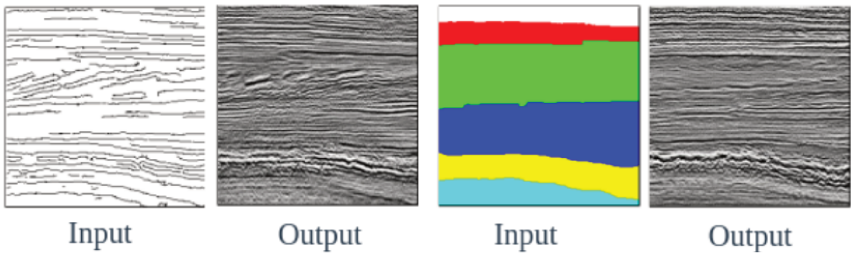


Figura 3.1: Exemplos de geração de dados sintéticos: os *inputs* representam os esboços de entrada da GAN, enquanto os *outputs* são os dados sintéticos gerados. Figura reproduzida de Ferreira et al. [19].

Wu et al. [74] propõem uma abordagem onde é aplicada uma maneira de criar diversas imagens sísmicas sintéticas. Para criar uma imagem sísmica, primeiro é gerado um modelo de refletividade horizontal $r = r(x, z)$ com uma sequência de valores de refletividade aleatórios que estão no intervalo de $[-1, 1]$. Em seguida, são realizadas algumas operações sobre o modelo, como por exemplo dobra *sinusoidal*, deslocamento vertical, cisalhamento linear. Além disso, são adicionadas falhas e, por fim, é feita a convolução dos modelos com um pulso do tipo Ricker [74] (descrito na Seção 4.1) com a frequência desejada (um exemplo pode ser visto na Figura 3.2). Entretanto, como esse gerador foi proposto para treinar uma rede de detecção de falhas, ele possui algumas limitações caso seja utilizado em outras tarefas, como por exemplo, a simplicidade geométrica do dado gerado e o fato de não possuir rótulos além do ângulo da falha.

Figura 3.2: Exemplos de geração de dados sintéticos: as imagens representam a sequência de passos realizados (da esquerda para a direita) para a geração de dados sintéticos. Figura reproduzida de Wu et al. [74].

3.2 Aplicações Sísmicas com GANs

Na aquisição e processamento sísmico, vários fatores podem provocar a escassez ou problemas nos dados, como por exemplo as restrições físicas do método — limitação no

comprimento disponível do receptor, problemas de gravação e iluminação do alvo. Isso pode resultar em problemas como traços ruidosos, lacunas na cobertura ou espaçamento irregular/inadequado, dificultando a interpretação geológica de uma área de interesse [47, 83].

Um método comumente utilizado para o aprimoramento dos dados é a interpolação. A interpolação de dados sísmicos ou traços sísmicos ausentes é um problema de longa data [52, 70]. Oliveira et al. [47] e Kaur et al. [32] exploram a interpolação de dados sísmicos utilizando a cGAN (*conditional* GAN) [44] e CycleGAN [85], respectivamente, e seus resultados superam os algoritmos tradicionais da literatura sísmica. Essas interpolações podem ajudar na interpretação geológica, aumentando a densidade espacial de uma seção sísmica e também podem ser usadas para reconstruir seções inteiras interpolando traços vizinhos, reduzindo os custos de campo [70]. Vale ressaltar que, em Oliveira et al. [47], o treinamento e o teste da rede são realizados em um mesmo cubo sísmico. Logo, os dados de treinamento e teste são potencialmente parecidos. Para uma avaliação mais confiável, os conjuntos deveriam ter sido obtidos de diferentes cubos sísmicos. Além disso, em ambas as técnicas, apesar da melhoria na resolução dos dados sísmicos, a presença de ruídos ainda é evidente. Além disso, Alwon et al. [2], Chang et al. [7], Gan et al. [20], Spitz [63] exploram com sucesso a interpolação de traços sísmicos ausentes ao invés da interpolação dos dados em si. Em Chang et al. [7], após o treinamento da SIGAN (*Seismic data Interpolation* GAN), o gerador é utilizado para reconstruir os traços sísmicos ausentes. De forma similar, Alwon et al. [2] desenvolveram a GANAN (GAN *Noise Attenuation Network).* Exemplos numéricos usando conjuntos de dados de campo terrestres e marinhos demonstram a validade e a eficácia dessas abordagens propostas baseadas em GANs [2, 7], ajudando assim o processo de interpretação.

A maioria das técnicas de interpretação é projetada para interpretar um certo padrão sísmico (por exemplo, falhas e domos de sal) em um conjunto de dados sísmicos, sendo este fundamental para representar a geologia subterrânea e auxiliar atividades em vários domínios, como engenharia ambiental e exploração de petróleo [13]. A melhoria destas técnicas vem sendo estudada recentemente com o uso de GANs [13, 39, 75].

Devido a restrições físicas ou financeiras, os conjuntos de dados sísmicos podem frequentemente ser sub-amostrados. Além disso, lacunas na cobertura, traços ruidosos e espaçamento irregular são problemas comuns em levantamentos terrestres e marítimos que podem dificultar a interpretação geológica de uma área de interesse. Logo, outro desafio ao se trabalhar com dados sísmicas é a aquisição de dados densamente amostrados [18]. No entanto, ter dados densos e regularmente amostrados tem se tornado cada vez mais importante no processamento sísmico, visto que há um crescente interesse nas CNNs em muitos trabalhos de pesquisa envolvendo imagens sísmicas [18]. Pensando nisso, Ferreira et al. [18] propõem a utilização da cGAN para o problema de reconstrução de dados em conjuntos de dados sísmicos pós-empilhamento. Além de aumentar a densidade espacial, as interpolações pós-empilhamento também podem ser usadas para reconstruir seções inteiras interpolando traços vizinhos. Siahkoohi et al. [59] também propõem algo similar utilizando GANs: a reconstrução de dados sísmicos, independentemente do tipo de amostragem, resultando na reconstrução de dados sísmicos com 10% de informação (veja Figura 3.3). Entretanto, os dados utilizados para o treinamento e para o teste (onde

os resultados do trabalho são reportados) são do mesmo dado sísmico 3D. Além disso, também é necessário fazer a rotulação manual dos exemplos para o treinamento da rede, isto é, fazer a reconstrução manual dos dados, o que é inviável quando consideramos a quantidade de dados necessários para treinar uma rede neural profunda.

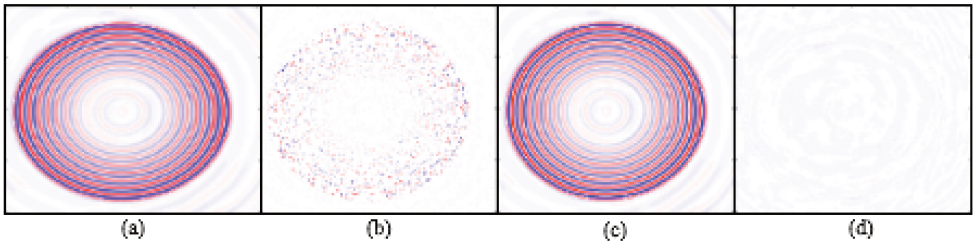


Figura 3.3: Exemplos de resultados de Siahkoohi et al. [59]: em (a) temos o dado original, de onde foram removidas regiões de forma aleatória gerando (b). Aplicando o gerador da GAN em (b) obtém-se o dado reconstruído (c). Por fim, em (d) temos a diferença entre o dado original e o dado gerado pela rede. Figura reproduzida de Siahkoohi et al. [59].

Alguns autores também exploram a detecção de falhas [39, 74, 75], visto que o mapeamento de planos de falha usando imagens sísmicas é uma etapa crucial na interpretação sísmica. Convencionalmente, isso requer esforços manuais significativos que normalmente passam por várias iterações para otimizar como os diferentes planos de falha se conectam. Muitas técnicas foram desenvolvidas para automatizar esse processo, como estimativa de coerência sísmica, detecção de bordas e rastreamento de falhas. No entanto, essas técnicas não aproveitam a experiência dos intérpretes. Nessa direção, Xiong et al. [75] apresentam um dos primeiros métodos que utiliza uma CNN para detectar e mapear automaticamente zonas de falha usando imagens sísmicas 3D de maneira semelhante à feita pelos intérpretes. Na sequência, Lu et al. [39] apresentam uma contribuição interessante para as redes de detecção de falhas ao desenvolver uma abordagem baseada em GAN para o pré-processamento de dados no domínio da imagem, visando aprimorar a imagem sísmica antes do treinamento (veja Figura 3.4). Tal abordagem, além de gerar imagens super-amostradas, preserva o conteúdo da frequência espacial e temporal dos dados originais, fornecendo um conjunto de dados mais denso para a CNN na detecção de falhas.

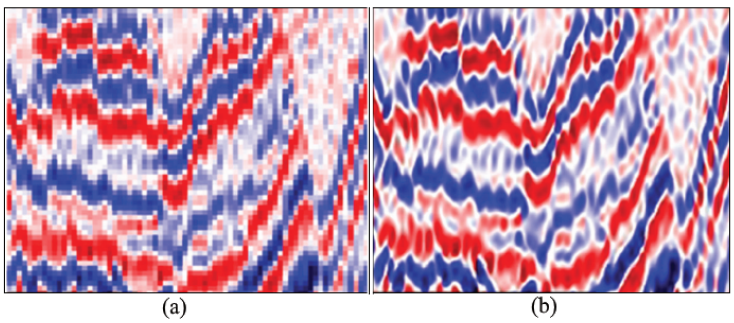


Figura 3.4: Exemplos de resultados de Lu et al. [39]: em (a) temos o dado original e em (b) temos o dado gerado pela GAN. Figura reproduzida de Lu et al. [39].

O aprimoramento de frequência é um outro método de aprimoramento de dados sísmicos. Nos métodos tradicionais, a resolução alcançável pelas técnicas de imagem sísmica padrão é limitada, em teoria, pela física da propagação de ondas subterrâneas e, na prática, pelo custo computacional associado à imagem em altas frequências [24]. A classe emergente de algoritmos de *Deep Learning* oferece oportunidades para abordar as duas limitações, estabelecendo relacionamentos entre versões de imagens sísmicas de baixa e alta resolução.

Entretanto, para solucionar as limitações de resolução e contaminação por ruído, Dutta et al. [16] propõem um método que realiza aprimoramento e remoção de ruídos em frequência utilizando GANs. Similarmente, Zhang et al. [83] apresentam um aprimoramento da largura de banda de frequência dos dados sísmicos utilizando uma cGAN [44]. Além disso, Halpert et al. [24] treinaram uma GAN para produzir imagens de alta resolução tendo como entrada imagens de baixa resolução. O resultado em dados sintéticos foram superiores aos métodos tradicionais da literatura sísmica.

Adicionalmente, vários autores como Lu et al. [40], Mosser et al. [45], Richardson [55], Yang e Ma [76] exploram a geração e melhora de modelos de velocidade utilizando GANs. Mais especificamente, Richardson [55] treina uma DCGAN [79] (*Deep Convolutional* GAN) para produzir modelos de velocidade realistas. Seus resultados sugerem que o gerador aprendeu a produzir modelos de velocidade plausíveis a partir de vetores latentes. Da mesma forma, Mosser et al. [45] apresentam modelos de velocidade realistas quando aplicados a um conjunto de dados real.

Em relação à remoção de ruídos de dados sísmicos, o trabalho que mais o nosso se aproxima é o de Ganssle [21]. O autor utiliza a GAN Pix2Pix, umas das redes na qual o nosso trabalho se baseou, e obtêm sucesso na remoção de ruído branco. Entretanto, o trabalho apresenta algumas limitações, como por exemplo, a simplicidade dos dados utilizados (tanto os sintéticos quanto os dados reais), a rede utilizada não contém adaptação para dados sísmicos, e os dados gerados pela rede aparentam visualmente ter perdido algumas informações estruturais.

Por fim, as GANs também estão presentes em outras aplicações dentro da exploração sísmica que não foram exploradas neste texto, como por exemplo, a transformação de uma imagem migrada para a respectiva imagem de refletividade [50] ou em análise de reservatórios [8, 81].

3.3 Resumo

As abordagens tradicionais baseadas na Física para inferir propriedades da superfície, no geral, são demoradas e computacionalmente caras [45]. Os trabalhos apresentados geram imagens sísmicas aprimoradas que auxiliam a reduzir as incertezas nos dados e, consequentemente, aprimoram a interpretação sísmica. Na Tabela 3.1, apresentamos um resumo dos trabalhos abordados neste capítulo, contendo a aplicação do trabalho, o modelo utilizado e ano de publicação. Também destacamos na tabela os trabalhos mais importantes para o desenvolvimento do nosso trabalho.

- O trabalho de Ferreira et al. [19] apresenta uma abordagem focada na geração de

dados sísmicos sintéticos, que foi um dos desafios neste trabalho.

- O trabalho de Wu et al. [74] fundamentou a construção das nossas bases sintéticas.
- Os trabalhos de Ganssle [21] e Picetti et al. [51] são os primeiros a utilizar a Pix2Pix (uma das GANs utilizadas no nosso trabalho), na Sísmica, sendo que o segundo trabalho também apresenta também a aplicação de remoção de ruídos.
- Os trabalhos de Dutta et al. [16] e Zhang et al. [83] apresentam diferentes técnicas para o aprimoramento de frequência, um dos objetivos do nosso trabalho.
- O trabalho de Lu et al. [39] apresenta uma metodologia similar à que desenvolvemos, onde é desejado manter as principais propriedades de dados sísmicos.

Considerando os trabalhos revisados, também selecionamos as técnicas para validação dos resultados, como *Structural Similarity* (SSIM), *Signal-to-Noise Ratio* (SNR), e validação com especialistas.

Outro desafio notado durante a revisão da literatura foi a avaliação das abordagens apenas com dados sintéticos. Apresentamos nesta dissertação os resultados em dados reais para a melhor análise dos resultados obtidos.

Como é possível analisar nos trabalhos apresentados, as GANs já fazem parte de vários processos na sísmica, e vem apresentando ótimos resultados [8, 24, 47, 81]. Por isso, escolhemos essas redes para ser utilizadas neste trabalho. No capítulo seguinte apresentaremos a metodologia para criação dos dados sintéticos, criação dos nossos modelos de GANs, treinamento e validação de resultados.

Tabela 3.1: Resumo dos trabalhos apresentados neste capítulo. Os trabalhos em destaque são aqueles que consideramos mais relevantes para o desenvolvimento deste trabalho. GAN: *Generative Adversarial Network*, CSA-GAN: *Compressed Sensing Architecture* GAN, cGAN: *Conditional* GAN, SIGAN: *Seismic data Interpolation* GAN, DCGAN: *Deep Convolutional* GAN, GANAN: GAN *Noise Attenuation Network.*

Aplicação	**Modelo**	**Referência**
Geração de dados sintéticos	GAN	Ferreira et al., 2019 [19]
Aquisição de dados sísmicos	CSA-GAN	Zhang et al., 2019 [82]
Processamento sísmico	GANAN	Alwon, 2018 [2]
Interpolação	cGAN	Oliveira et al., 2019 [47]
Interpolação	SIGAN	Chang et al., 2018 [7]
Reconstrução de dados sísmicos	GAN	Siahkoohi et al., 2018 [59]
Reconstrução de dados sísmicos	cGAN	Ferreira et al., 2019 [18]
Resolução sísmica	cGAN	Chen et al., 2018 [8]
Resolução sísmica	GAN	Halpert, 2018 [24]
Resolução sísmica	Pix2Pix	Picetti et al., 2019 [51]
Aprimoramento de frequência	GAN	Dutta et al., 2019 [16]
Ganho de frequência	CNN	Zhang et al., 2019 [83]
Remoção de ruídos	Pix2Pix	Ganssle, 2018 [21]
Aplicações de imagem sísmica	GAN	Picetti et al., 2018 [50]
Falhas	DCGAN	Lu et al., 2018 [39]
Falhas	CNN	Wu et al., 2018 [74]
Redução de ordem modelo	GAN	Richardson, 2018 [55]
Modelo de velocidade	DCGAN	Mosser et al., 2018 [45]
Previsão de litologia sísmica	CNN	Zhang et al., 2018 [81]

Capítulo 4

Metodologia Proposta

Neste capítulo, apresentamos nossa abordagem baseada em GANs para a remoção de ruídos, ganho de alta frequência e mantendo a integridade dos dados sísmicos. A Figura 4.1 ilustra uma visão geral da nossa proposta. A parte (a) corresponde ao treinamento da GAN e pode ser descrito da seguinte forma: o gerador, $G : \mathbb{X} \to \mathbb{X}$ (ou $G : \mathbb{X} \times \mathbb{X} \to \mathbb{X}$ para Pix2Pix), recebe o dado ruidoso como entrada e traduz para um dado sem ruído. Assim, $G(X)$ (ou $G(X, X)$), $Y \in \mathbb{X}$ alimentam o discriminador, $D : \mathbb{X} \times \mathbb{X} \to [0, 1]$, que classifica os dados como reais (dados do conjunto de treinamento) ou falsos (dados gerados). A parte (b) corresponde à avaliação dos resultados e pode ser descrita da seguinte forma: computamos a diferença

$$\delta = Y - G(X), \tag{4.1}$$

juntamente com algumas métricas (acurácia ACC, *Signal-to-Noise Ratio* (SNR), e *Structural Similarity Index* (SSIM)) descritas a seguir.

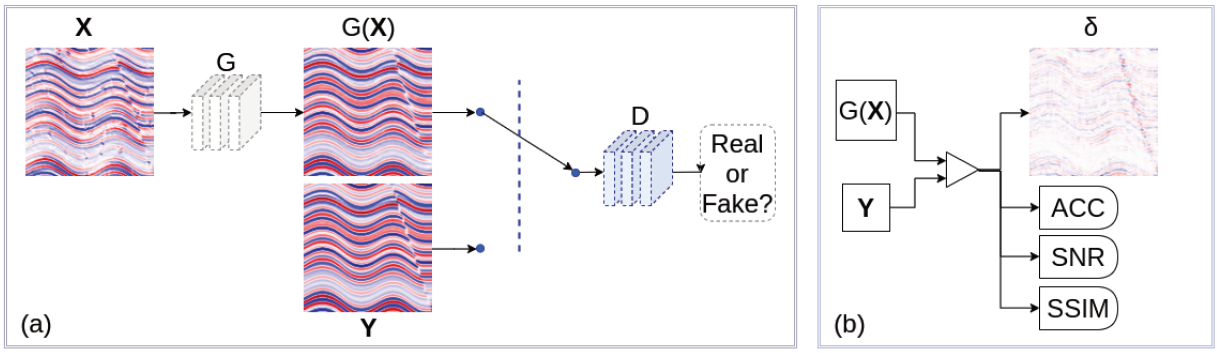


Figura 4.1: Visão geral da abordagem proposta baseada em GANs para a remoção de ruídos e ganho de alta frequência, mantendo a integridade dos dados sísmicos. A parte (a) corresponde ao treinamento da GAN e a parte (b) corresponde à avaliação dos resultados.

Para o desenvolvimento da nossa abordagem, temos dois passos principais. O passo inicial refere-se aos dados que serão utilizados para realizar o treinamento dos modelos, pois, assim como nos modelos de *Deep Learning* em geral, é necessário um conjunto de dados representativos do problema que desejamos resolver. Logo, o primeiro passo

consiste na criação de um gerador de imagens sísmicas sintéticas com e sem ruídos. O desenvolvimento do gerador é apresentado na Seção 4.1 deste capítulo.

O segundo passo consiste na seleção e adaptação de arquiteturas de GANs com características similares ao nosso problema (Seção 4.2). A primeira arquitetura que selecionamos foi a Pix2Pix [29], devido aos seus bons resultados na tarefa de tradução de imagem para imagem e por ter sido a primeira com tal proposta na literatura. Fizemos algumas alterações, conforme apresentamos na Seção 4.2.1, e treinamos o nosso *baseline* utilizando dados sísmicos convertidos para imagens em tons de cinza. Em seguida, trabalhamos na seleção de novas arquiteturas e em modificações para realizar o treinamento com dados sísmicos brutos ao invés de imagens. Para isso, selecionamos mais duas arquiteturas, seguindo o estado da arte de GANs: Pix2PixHD [71] e SPADE [49]. Adaptamos ambas as redes para o nosso problema, conforme apresentado na Seção 4.2.2 e na Seção 4.2.3, respectivamente. Essas redes foram selecionadas devido à rede Pix2Pix ter apresentado um bom resultado, e essas outras terem sido desenvolvidas baseadas nela.

Em resumo, desenvolvemos três abordagens baseadas em GANs para remoção de ruídos e ganho de alta frequência: uma versão modificada da Pix2Pix (somente para remoção de ruídos), uma versão modificada da Pix2PixHD, que é uma rede Pix2Pix aprimorada para geração de imagens de alta resolução e uma versão modificada da SPADE.

Adicionalmente, desenvolvemos novas funções de custos e incorporamos funções utilizadas na literatura (tanto da Computação, quanto da Sísmica) às novas redes que desenvolvemos, com o objetivo de melhorar nossos resultados (Seção 4.3). Também criamos novas funções para a avaliação dos resultados, além de utilizarmos as métricas padrão da literatura (Seção 4.4).

Por fim, após a obtenção de resultados satisfatórios em dados sintéticos — isto é, avaliados e aprovados por especialistas e com valores de acurácia superiores a 70% —, aplicamos o nosso modelo treinado em dados reais, disponibilizados pelo laboratório *High-Perfomance Geophysics* (HPG) situado no Centro de Estudo de Petróleo (CEPETRO) da UNICAMP, finalizando o fluxo de desenvolvimento previsto para este trabalho.

Para facilitar o entendimento e a aplicação deste Mestrado, descrevemos no Apêndice A os fluxogramas de dados, treinamento, avaliação dos resultados, e utilização do modelo.

4.1 Gerador de Dados

Conforme apresentado na Seção 2.1, o treinamento de um modelo GAN requer um conjunto de dados rotulados, em que cada amostra do conjunto de dados é composta pelo par $p_{\text{data}} \times p_{\text{noise}}$, em que p_{data} e p_{noise} são os dados sem ruído ou *target* e os ruidosos, respectivamente. Portanto, para atenuar o problema de escassez de dados, propusemos um gerador de dados, que pode ser dividido em três etapas, a saber:

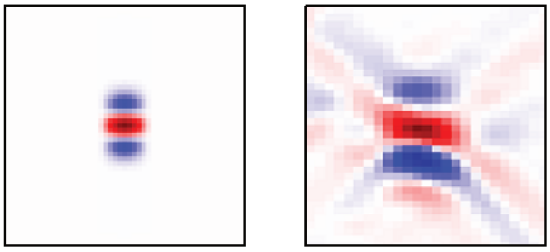
(i) Criação de estruturas geológicas: A primeira etapa para gerar os dados consiste na criação de estruturas geológicas representativas dos dados sísmicos migrados. Essas estruturas serão utilizadas para modelar tanto os dados com ruído quanto os sem ruído. A criação de uma estrutura que seja representativa de dados reais é um dos desafio deste trabalho.

(ii) Geração de dados sem ruído: Para modelarmos dados sem ruído, utilizamos a *wavelet Ricker* — pulso matemático sintético — que é capaz de representar o modelo de uma aquisição sísmica perfeita. Ele é frequentemente empregado para modelar dados sísmicos [26], e pode ser definido como:

$$A(t; f) = \left(1 - 2\pi^2 f^2 t^2\right) \mathrm{e}^{-\pi^2 f^2 t^2}, \tag{4.2}$$

onde A é a amplitude do sinal com frequência de pico f e tempo t [78]. Para a geração de dados com ganho de alta frequência, modificamos f para a frequência objetivo. Então, convolvemos as estruturas com a wavelet Ricker (veja a Figura 4.2a) e geramos os dados sem ruído.

(iii) Geração de dados ruidosos: A representação dos dados ruidosos deve aproximar-se dos dados reais, tendo em vista que na aquisição também são capturadas interferências produzidas por fenômenos da natureza e da própria onda que foi introduzida. Para isso, modelamos os dados através da convolução das estruturas com uma função de dispersão pontual (PSF, *Point Spread Function*). Esta função pode ser usada para descrever o grau de desfoque e degradação em uma imagem obtida de um objeto alvo como uma resposta de um sistema de imagem [46]. Para que nossos dados possuam características de um dado real, escolhemos PSFs extraídas de dados sísmicos reais via um processo de extração de difrações — em um dado de difração já estão presentes algumas PSFs, sendo necessário apenas realizar sua extração. Então, para cada estrutura foi aplicada uma convolução com uma PSF. As PSFs são escolhidas aleatoriamente do nosso conjunto. Na Figura 4.2b, apresentamos um exemplo.



(a) Ricker (b) PSF

Figura 4.2: Uma Ricker Wavelet é um pulso matemático sintético que pode representar uma aquisição sísmica perfeita. As PSFs podem ser extraídas de dados reais através da migração de difratores sísmicos.

Utilizando essa metodologia, geramos duas versões diferentes do gerador de dados, conforme apresentado nas seções seguintes. A versão 1 (GD1) foi a nossa primeira versão do gerador que foi validada pelos especialistas[1]. Para a versão 2 (GD2), fizemos algumas melhorias considerando os conhecimentos adquiridos no decorrer dos experimentos realizados e dos novos trabalhos da literatura. Neste contexto, o GD2 contempla tudo que temos no GD1, exceto o ganho de altas frequências.

[1]Por especialistas, queremos dizer que foi avaliado por no mínimo 2 geofísicos e que houve concordância entre eles.

Gerador de Dados – Versão 1 (GD1)

Como primeira arquitetura de criação de modelos para nosso target, utilizamos funções que simulam estruturas geológicas (alguns exemplos são apresentados na Tabela 4.1 — as funções foram desenvolvidas de forma empírica, de acordo com a experiência dos especialistas e o trabalho de Wu et al. [74]), onde os valores para interfaces refletivas são escolhidos a partir de uma distribuição uniforme definida entre $[-1, 1]$. Em seguida, adicionamos falhas sísmicas de acordo com o trabalho de Wu et al. [74]. Para gerar o dado sem ruído, fazemos a convolução dessa estrutura com a Ricker, enquanto que para gerar o dado ruidoso, primeiro adicionamos ruídos confettis e gaussiano, e então fazemos a convolução com a PSF.

Tabela 4.1: Funções usadas para geração de dados sintéticos.

(1)	$f_1(x) = \beta x + \eta \sin(\theta x)$
(2)	$f_2(x) = \beta x + \eta \sin((\theta x)^2)$
(3)	$f_3(x) = \alpha x^2 + \beta x + \gamma$
(4)	$f_4(x) = \beta x + \eta \cos(\theta x)$
(5)	$f_5(x) = \nu\sqrt{1 - (\xi x)^2}$
(6)	$f_{12}(x) = f_1(f_2(x))$
(7)	$f_{21}(x) = f_2(f_1(x))$
(8)	$f_{31}(x) = f_3(f_1(x))$

Entretanto, após o aprofundamento da pesquisa e estudo mais aprimorado sobre geração de dados, percebemos que o processo, GD1, para criação dos dados sísmicos não era suficiente para o nosso trabalho. Além disso, o GD1 não tem a opção de representar estruturas do tipo múltiplas falhas e sal. Portanto, construímos uma segunda versão, desenvolvida juntamente com HPG/CEPETRO, com uma metodologia de geração de estruturas diferentes visando obter dados mais realistas, conforme apresentado a seguir.

Gerador de Dados – Versão 2 (GD2)

A geração das estruturas possuem alguns passos adicionais, ao compararmos com a GD1, sendo que alguns deles são baseados nos trabalhos de Wu et al. [74] e Geng et al. [22], e alguns trabalhos desenvolvidos pelo HPG, conforme listado a seguir.

- Geração de estruturas planas (ao contrário da versão anterior que utilizava funções);
- Adição de região de sal, de acordo com a probabilidade $p1$: consiste de um polígono com quantidade de lados e ângulos aleatórios;
- Deformação por dobramento: as interfaces são dobradas seguindo uma composição de senoides com decaimento (com parâmetros aleatórios);
- Deformação por cisalhamento: é aplicada uma transformação geométrica de cisalhamento (com parâmetros aleatórios);

- Deformação por falha, de acordo com a probabilidade $p2$: é aplicada uma transformação que imita o comportamento de uma falha sísmica natural, com o deslocamento das interfaces ao longo de uma reta (parâmetros aleatórios);
- Deformação por falha, de acordo com a probabilidade $p3$: se a probabilidade $p2$ for sorteada, e $p3 < p2$, uma nova deformação por falha é realizada;
- As probabilidades $p1$, $p2$ e $p3$ são parâmetros configuráveis.

Por fim, tanto o dado sem ruído quanto o ruidoso seguem a mesma estrutura de aplicação ocorrida no GD1 da subseção anterior. Com isso, temos um novo gerador mais robusto, capaz de gerar dados mais próximos dos reais devido às mudanças na geração das estruturas geológicas e adição de novas informações, como a de sal, por exemplo.

4.2 GANs para a Remoção de Ruídos

Nesta seção, apresentamos as arquiteturas utilizadas, assim como as modificações necessárias para a remoção de ruídos e ganho de altas frequências.

4.2.1 Pix2Pix

A Pix2Pix [29], primeira rede utilizada neste trabalho, foi originalmente proposta para tradução de imagens para imagens. Nessas condições, implementamos um conversor de dados sísmicos para imagens. Os dados gerados pelo nosso gerador são convertidos para uma imagem em escala de cinza (uma vez que os dados sísmicos possuem apenas um canal), que é então repassada para a rede. Do mesmo modo, a saída da rede é uma imagem. Então, implementamos uma conversão da imagem em escala de cinza para dados sísmicos.

Em termos práticos, para treinar a Pix2Pix o dado sísmico é convertido para imagem, repassado para a rede, que gera uma imagem de saída, essa imagem é normalizada entre $[-1, 1]$ e convertida de volta para dados sísmicos. Tais passos adicionais não têm impacto no tempo e nos resultados do processo de treinamento desta rede, visto que as camadas não foram alteradas.

4.2.2 Pix2PixHD

Inicialmente, as mesmas alterações feitas na Pix2Pix (conforme Seção 4.2.1) também foram feitas na Pix2PixHD [71], sendo adicionadas:

- A normalização para o valor máximo absoluto dos dados, ou seja, $[-\sigma, \sigma]$, onde σ é definido por:

$$\sigma = \max\{|X_{ij}|, |Y_{ij}|, |G(X)_{ij}|\} \ . \tag{4.3}$$

  Desse modo, todos os dados são normalizados em uma mesma escala, permitindo que as diferenças (ou seja, os erros) possam ser analisados de forma mais assertiva;

- A Pix2PixHD usa a função de custo *perceptual* [31] que requer uma imagem com três canais e, como os dados sísmicos têm apenas um canal, nós replicamos os dados nos três canais para calcular essa função de custo. Portanto, nosso domínio das imagens, $\mathbb{X}$, são de $n_X \times m_X$.

Após o treinamento do modelo e comparação com a Pix2Pix, iniciamos as novas alterações nessa rede, como descrito a seguir.

- Modificamos a entrada da rede para dados sísmicos brutos, fornecidos diretamente de nosso gerador, eliminando a etapa de conversão de imagem em escala de cinza, o que agrega ganho de informação para a rede;

- A saída da rede (isto é, o dado de saída do gerador) era normalizada, convertida para inteiros (0 a 255) e salvos como imagens. Removemos todos esses passos e salvamos dados sísmicos brutos (isto é, matriz de pontos flutuantes com precisão de 32 *bits* que pode ser facilmente convertida para *Seismic Unix* (SU), por exemplo);

- Operações de pre-processamento de dados são comumente utilizadas na grande maioria das redes profundas atuais, o que também se aplica a Pix2PixHD. Operações tais como, normalização entre um faixa, conversão de tipo de dados, *scale* de dados, *resize*, *flip*, *crop*, zoom, etc. Neste contexto, foram removidos todos os pré-processamentos, normalizações e transformações geométricas da Pix2PixHD original. Isso foi necessário devido às características dos dados sísmicos e à necessidade de manter suas propriedades;

- Implementamos o cálculo de acurácia de treinamento (conforme Seção 4.4) que apesar de não ser utilizada para reportar resultados, ajudou no processo de seleção dos hiperparâmetros e aprimoramento dos modelos.

4.2.3 SPADE

Dado que as alterações realizadas na Pix2PixHD apresentaram melhorias significativas nos resultados, realizamos as mesmas alterações na SPADE [49], incluindo a modificação necessária para a utilização da função de custo *perceptual*. Entretanto, algumas modificações adicionais também foram necessárias para que fosse realizado o treinamento desta rede utilizando dados sísmicos.

Isso se deve ao fato desta rede ter sido projetada para tradução de mapas semânticos para imagens coloridas, não sendo capaz de produzir imagens de saída com apenas um canal de cor. Logo, substituímos a última camada convolucional para gerar dados com um canal ao invés de três. Além disso, a SPADE cria um mapa de rótulo *one-hot* a partir dos dados de entrada (que originalmente era mapas semânticos em que cada região é dividida por cores) e usava esse novo mapa como entrada da rede. Removemos todas as conversões e modificamos a entrada para dados sísmicos e os repassamos diretamente para a rede.

4.3 Funções de Custo

Neste trabalho, também implementamos funções de custo (*loss functions*) adicionais em nossa rede baseada na Pix2PixHD que foram combinadas com as outras funções de custo desta rede. Portanto, da Equação 2.7, temos nossa função de custo a seguir,

$$\begin{aligned}\mathcal{L}_{\text{cost}}(G_*, \{D_k\}_*) &= \min_G \left(\left(\max_{D_1,D_2,D_3} \sum_{k=1,2,3} \mathcal{F}_{\text{HD}}\left(G, D_k\right) \right) \right. \\ &+ \left. \lambda_1 \sum_{k=1,2,3} \mathcal{F}_{\text{FM}}\left(G, D_k\right) + \lambda_2 \mathcal{F}_{\text{P}}(G) + \lambda_3 \mathcal{F}_{\text{loss}}\left(G\right) \right), \end{aligned} \tag{4.4}$$

onde para λ_1 e λ_2 foram utilizados os valores propostos por Wang et al. [71] e $\mathcal{F}_{\text{loss}}(G)$ será definida de acordo com as medidas a seguir.

4.3.1 SEISMIC

A primeira função de custo que desenvolvemos foi a SEISMIC que retorna um valor entre 0 e 1 de acordo com o quão perto os dados gerados estão dos dados sem ruído e é definida na Equação 4.5 em que $||\cdot||_F$ é a norma de *Frobenius* [68], definida como

$$\text{SEISMIC}(X, Y) = \frac{||Y - G(X)||_F}{||Y||_F + ||G(X)||_F}\,. \tag{4.5}$$

Portanto, da Equação 4.4 e chamando $\mathcal{F}_{\text{loss}}(G)$ por $\mathcal{F}_{\text{SEIMIC}}(G)$ com tal função de custo definida temos

$$\mathcal{F}_{\text{SEISMIC}}(G) = E_{(X\sim p_{\text{noise}}; Y\sim p_{\text{data}})}[\text{SEISMIC}(X, Y)]\,. \tag{4.6}$$

4.3.2 SSIM e MSSSIM

A próxima função de custo desenvolvida teve como base o *Structural Similarity Index* (SSIM) [9] (Equação 4.7), que mede a diferença perceptual entre dois dados similares. Essa métrica mostrou-se eficaz para validar resultados e melhorar o modelo quando usada como função de custo, uma vez que o SSIM foi desenvolvido para melhorar os métodos tradicionais, como o *Peak Signal-to-Noise Ratio* (PSNR) e *Mean Squared Error* (MSE). A função de custo $\mathcal{F}_{SSIM}$ pode ser definida conforme a Equação 4.8. Além disso, implementamos uma outra função de custo semelhante, com base na *Multiscale* SSIM (MSSSIM) [72], que é uma versão mais avançada da SSIM, onde o cálculo é realizado em várias escalas através de subamostragem. Na prática, é aplicado o filtro do tipo *downscaling*, em cada iteração do cálculo, nos dois sinais recebidos o que reduz a amostragem da imagem filtrada por um fator de 2. Esse método fornece mais flexibilidade do que os métodos de escala única ao incorporar as variações de resolução da imagem [72].

$$\text{SSIM}(X, Y) = \frac{(2\mu_X\mu_Y + C_1) + (2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)}, \tag{4.7}$$

onde μ_X é a média de $G(X)$, μ_Y é a média de Y, σ_X^2 é a variância de $G(X)$, σ_Y^2 é a variância de Y, σ_{XY} é a covariância de $G(X)$ e Y, C_1 e C_2 são variáveis para estabilizar a divisão com o denominador. Neste trabalho, os valores de C_1 e C_2 foram computados como $(k_1 L)^2$ e $(k_2 L)^2$, respectivamente, onde L é a faixa dos valores de píxeis (a diferença entre o maior valor e menor valor — normalmente $2^{\#bits\ por\ pixels} - 1$) e $k1 = 0.01$ e $k2 = 0.03$, conforme proposto por Wang et al. [72]. Para nossas funções de custo, $\mathcal{F}_{\text{loss}}$, definimos $\mathcal{F}_{\text{SSIM}}$ e $\mathcal{F}_{\text{MSSSIM}}$ como

$$\mathcal{F}_{\text{SSIM}}(G) = E_{(X\sim p_{\text{noise}};Y\sim p_{\text{data}})}[1 - \text{SSIM}(X,Y)]\,; \tag{4.8}$$

$$\mathcal{F}_{\text{MSSSIM}}(G) = E_{(X\sim p_{\text{noise}};Y\sim p_{\text{data}})}[1 - \text{MSSSIM}(X,Y)]\,. \tag{4.9}$$

4.3.3 Fourier

Para ajudar a manter as propriedades de frequência dos dados gerados, desenvolvemos uma função de custo que penaliza a diferença de frequências entre a saída da rede e o dado sem ruído (*target*). Essa função de custo pode ser definida pela Equação 4.10, em que SSIM representa uma função que computa a similaridade ou distância entre dois dados, como por exemplo, no espectro de Fourier, e $\mathcal{F}_2$ como a transformada bidimensional de Fourier.

$$\mathcal{F}_{\text{FREQ}}(G) = E_{(X\sim p_{\text{noise}};Y\sim p_{\text{data}})}[\text{SSIM}(\mathcal{F}_2(G(X)), \mathcal{F}_2(Y))]. \tag{4.10}$$

4.3.4 Normal

Finalmente, experimentamos a função de custo Normal introduzida por Geng et al. [22]. Os autores a propuseram para medir a precisão das normais da superfície de um volume de Tempo Geológico Relativo em relação a sua verdade básica. Esta função de custo computa a derivada local e, a partir dela, computa a normal local sendo, então, extraída a medida de similaridade do cosseno das duas normais[2]. A normal da curva é mais sensível a pequenas mudanças do que a derivada o que tornaria esta função de custo mais sensível à orientação das estruturas geológicas [22]. Modificadas as variáveis para o nosso contexto, a superfície normal dos dados gerados $g(x_1, x_2) = G(X)$ e dos dados originais $y(x_1, x_2) = Y$ podem ser denotados por

$$\mathbf{g}_{ij} = \left(-\frac{\partial g}{\partial x_1}, -\frac{\partial g}{\partial x_2}, 1\right)\Bigg|_{(x_1,x_2)_{ij}}, \tag{4.11}$$

$$\mathbf{y}_{ij} = \left(-\frac{\partial y}{\partial x_1}, -\frac{\partial y}{\partial x_2}, 1\right)\Bigg|_{(x_1,x_2)_{ij}}, \tag{4.12}$$

[2] `https://en.wikipedia.org/wiki/Cosine_similarity`

respectivamente. A função de custo Normal pode ser definida como

$$\mathcal{F}_{\text{NORMAL}}(X,Y) = E_{(X\sim p_{\text{noise}};Y\sim p_{\text{data}})}\left[1 - \frac{1}{n_X m_X}\sum_{j=1}^{n_X}\sum_{i=1}^{m_X}\left\langle \frac{\mathbf{g}_{ij}}{\|\mathbf{g}_{ij}\|_2}, \frac{\mathbf{y}_{ij}}{\|\mathbf{y}_{ij}\|_2}\right\rangle\right], \quad (4.13)$$

onde $\langle\cdot,\cdot\rangle$ representa o produto interno entre dois vetores e $\|\cdot\|_2$ é a norma Euclidiana do vetor. Em [22], os autores mencionam que, para esta função de custo fornecer um ganho de desempenho para a rede, devem ser utilizados valores de λ_3 (fator de peso da função de custo dentro da função de custo total) pequenos em relação às demais funções de custos da rede.

4.4 Medidas de Avaliação

A análise da qualidade de dados sintéticos ainda é um problema que está em aberto. Medidas tradicionais, como a norma L_2, não avaliam as estatísticas dos resultados e, portanto, não mensuram as perdas estruturadas dos dados [29]. Esse problema se torna ainda mais complexo nos dados sísmicos, pois além dos dados gerados terem que ser visualmente similares aos dados de treinamento, também devem ser precisos e manter as informações geológicas.

Durante o treinamento da GAN, é realizada a otimização da função de custo total, que pode ser dividida em três partes: (i) a função de custo da GAN (*minmax*); (ii) as demais funções de custo específicas de cada uma das redes investigadas neste trabalho; (iii) e uma combinação das nossas próprias funções de custo. Entretanto, a função da GAN (*minmax*) ainda é uma das funções dominante dentro do treinamento e, visto que ela tem uma perda relativa, não podemos utilizar os seus valores de custos para mensurar a qualidade do treinamento ou dos resultados. Com isto, surge a necessidade da utilização de medidas de avaliação adicionais, tanto no treinamento, quanto na validação de dados preditos. Para tal propósito, implementamos as seguintes medidas:

- A acurácia dada pela seguinte equação:

$$ACC = (1 - \text{SEISMIC}(X,Y))\%. \quad (4.14)$$

- Para a validação visual dos resultados, será realizada a diferença elemento por elemento entre o dado sintético gerado (*output*) e o dado sintético sem ruído (*target*), conforme realizado por Siahkoohi et al. [59] para a validação de seus resultados durante o processo de recuperação de dados sísmicos. Em resumo, isso representa o resíduo entre o destino (Y) e os dados de saída ($G(X)$) obtidos de δ. Isso também nos permite observar o erro pelas intensidades das amplitudes;

- Para verificar a relação do sinal de ruído dos dados gerados, computamos a Relação Sinal-Ruído (SNR). Isso é essencial para saber quantas informações aleatórias existem no conjunto de dados de entrada e saída;

- Para medir a similaridade entre duas imagens, usamos o método *Structural Similarity* (SSIM), que também foi utilizada por Li et al. [37] para validar seus modelos de velocidades gerados em relação aos modelos de velocidade básicos. No treinamento, usamos essa medida para verificar a similaridade entre o dado gerado e o dado alvo.

Por fim, nosso principal meio de validação dos dados gerados será a análise por especialistas em Geofísica e a utilização dos dados em atividades práticas realizadas por eles.

Quanto as métricas quantitativas optamos por utilizar a acurácia como critério de comparação entre os nossos modelos, pois é a métrica mais apropriada de acordo com as avaliações realizadas. Apesar da métrica SSIM ser essencial para verificar as similaridades entre os dados, pequenas variações na SSIM podem não representar a qualidade de um modelo em relação a outro. Para a SNR, a variação pode ser muito grande entre os experimentos, visto que não estamos trabalhando com uma frequência fixa.

Capítulo 5

Experimentos e Resultados

Neste capítulo, apresentamos os experimentos realizados e os resultados alcançados durante o desenvolvimento deste Mestrado.

Na Seção 5.1, apresentamos os experimentos relacionados ao gerador de dados sísmicos sintéticos. Com os dados sísmicos sintéticos construídos, na Seção 5.2 apresentamos os experimentos com as GANs Pix2Pix (Seção 5.2.1), Pix2PixHD (Seção 5.2.2) e SPADE (Seção 5.2.3), utilizando o GD1. Um sumário dos resultados é apresentado na Seção 5.2.4.

Em seguida, considerando o melhor modelo obtido na Seção 5.2, realizamos os próximos experimentos com o GD2 (Seção 5.3). Também avaliamos diferentes funções de custo (Seção 5.4). Por fim, na Seção 5.5 validamos os nossos modelos no *Sigsbee*, um dado sintético conhecido na literatura, e em dados reais.

5.1 Gerador de Dados

Na Seção 5.1.1, apresentamos o processo experimental de extração das funções de dispersão pontual (*Point Spread Functions*, PSFs) utilizadas para geração dos dados sísmicos sintéticos. Na Seção 5.1.2, apresentamos os experimentos preliminares que foram executados com o intuito de familiarização com a estrutura dos dados sísmicos. Nas Seções 5.1.3 e 5.1.4, apresentamos, respectivamente, os resultados da versão 1 do gerador de dados (GD1) e da versão 2 (GD2).

5.1.1 Extração das PSFs

As PSFs utilizadas neste trabalho foram extraídas de dados sísmicos reais de difração para ser mais representativos de dados reais. Para isso, é necessário utilizar um dado de difração migrado. Extraímos essas PSFs manualmente do seguinte modo: um especialista seleciona o ponto central de cada umas das PSFs e então extraímos a região dessa PSF com uma margem de 20 pontos, ou seja, com dimensão 41×41[1]. Na Figura 5.1, apresentamos um dado sísmico de difração da bacia do Jequitinhonha, e algumas PSFs extraídas desse dado (região selecionada).

[1] A dimensão 41×41 foi definida com base em observações empíricas realizadas por especialistas.

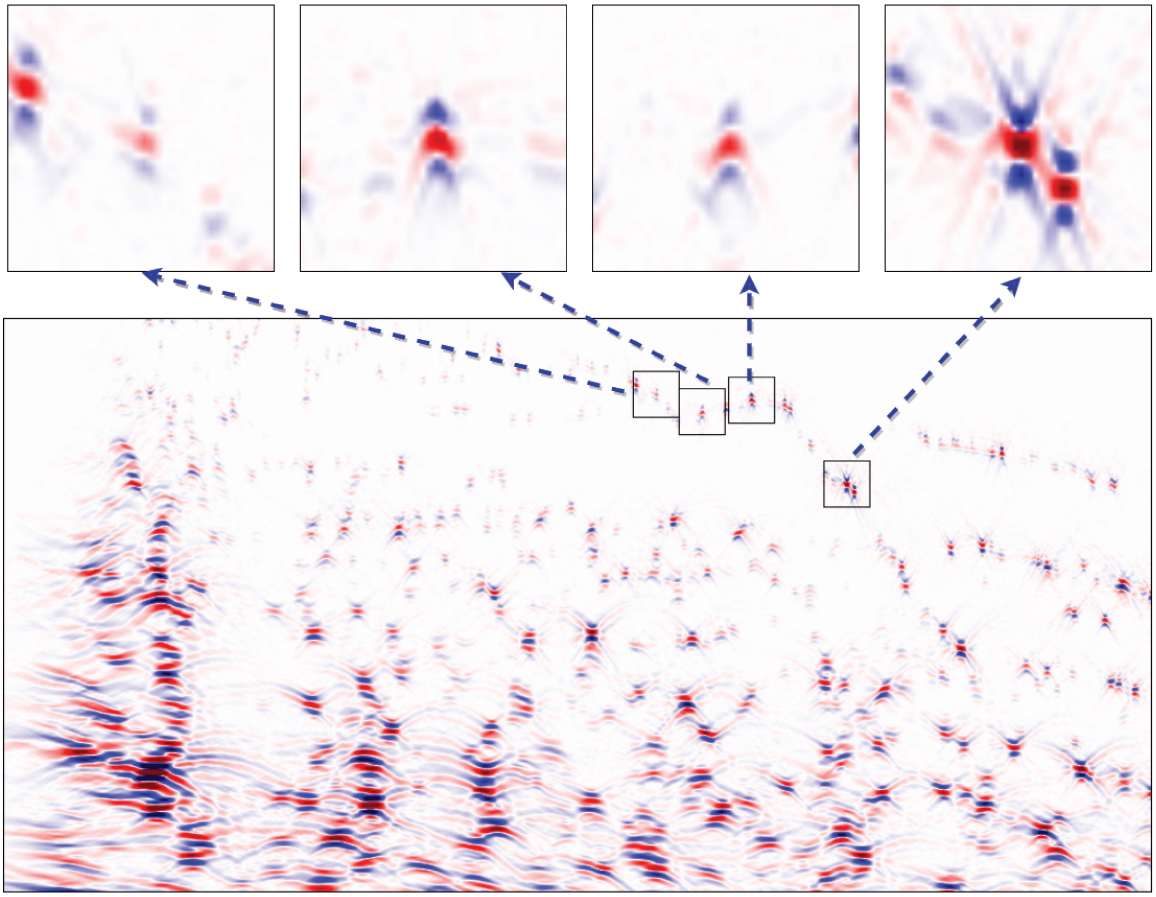


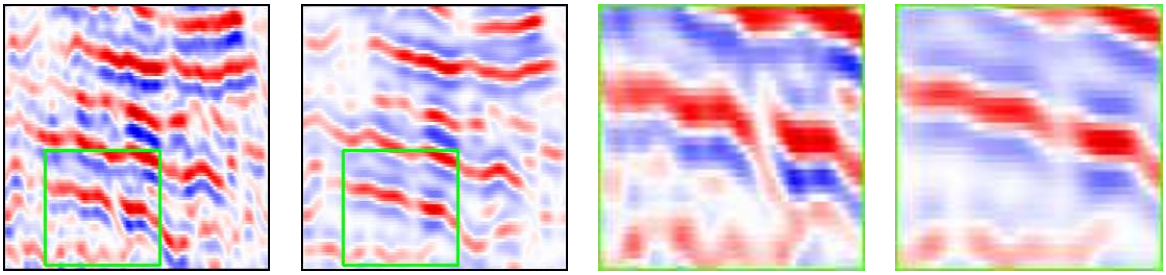
Figura 5.1: Dado de difração sísmica: o eixo vertical representa o tempo e o horizontal o ponto comum em profundidade (*Common Depth Point*, CDP). As regiões selecionadas representam algumas PSFs extraídas desse dado.

Além disso, após a extração da PSF, computamos sua frequência aproximada e desse modo podemos gerar uma Ricker 2D equivalente, isto é, com a mesma frequência e tamanho. Para computar a Ricker 2D, geramos a Ricker 1D e aplicamos um afinamento das bordas (*tapering*) usando a função cosseno, gerando assim uma Ricker 2D também de dimensão 41×41.

5.1.2 Dados Sísmicos: Experimentos Preliminares

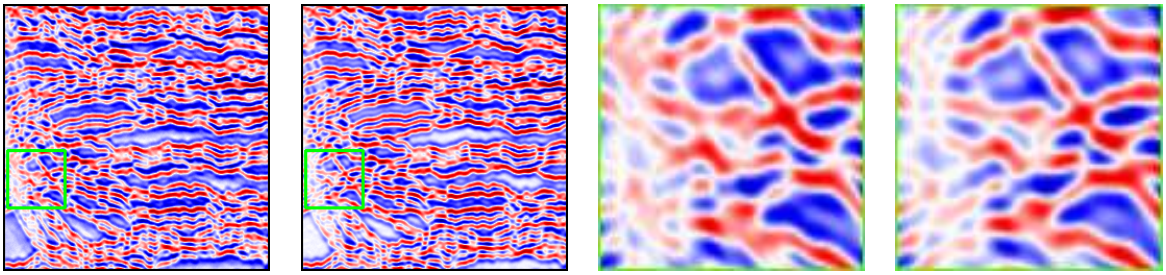
Para entender a estrutura dos dados sísmicos e o comportamento da Pix2Pix com estes dados, construímos dados sísmicos não avaliados por especialistas. Assim, geramos algumas estruturas de duas maneiras: 1) utilizando algoritmos de detecção de arestas em dados sísmicos reais, e 2) de forma aleatória. Em seguida, convolvemos estas estruturas com uma PSF e a Ricker para gerar o dado ruidoso e o dado sem ruído, respectivamente. Na Figura 5.2 apresentamos uma amostra gerada utilizando as estruturas baseadas em dados reais, enquanto na Figura 5.3 foi baseada em estruturas aleatórias.

Utilizando esses métodos, geramos 10.000 pares de imagens para o treinamento e 2.000 para validação, com resolução de 256×256. Treinamos a Pix2Pix com esses dados por 800 épocas (aproximadamente 63 mil iterações) por 24 horas em uma GPU NVIDIA



(a) (b) (c) (d)

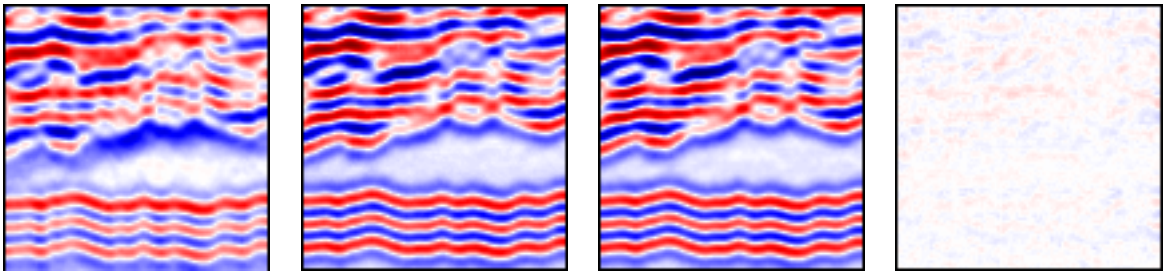
Figura 5.2: Amostra sintética gerada com uma estrutura extraída de dados reais: (a) apresenta a imagem ruidosa e (b) apresenta a imagem sem ruído. Em (c) e (d) apresentamos uma aproximação da região em destaque em (a) e (b), respectivamente, para demostramos uma das diferenças entre a convolução com a Ricker e a PSF.



(a) (b) (c) (d)

Figura 5.3: Amostra sintética gerada com uma estrutura gerada de forma aleatória: (a) apresenta a imagem ruidosa e (b) apresenta a imagem sem ruído. Em (c) e (d) apresentamos uma aproximação da região em destaque em (a) e (b), respectivamente.

Tesla V100[2]. Um exemplo do resultado desse treinamento pode ser visto na Figura 5.4.



(a) (b) (c) (d)

Figura 5.4: Treinamento com a Pix2Pix: (a) imagem ruidosa; (b) imagem sem ruído; (c) imagem gerada pela rede (d) diferença entre a imagem alvo e a imagem gerada.

Destacamos que, nesse experimento inicial, os dados sísmicos sintéticos não representam adequadamente os dados reais. Além disso, os resultados não foram validados por especialistas em Geofísica. Utilizamos esse experimento para verificar a viabilidade da pesquisa e demonstrar que as GANs são capazes de gerar dados sísmicos.

[2]https://www.nvidia.com/en-us/data-center/v100

5.1.3 Gerador de Dados – Versão 1 (GD1)

Para gerar as estruturas geológicas, usamos uma das funções apresentadas na Tabela 4.1 (escolhida aleatoriamente). O painel foi gerado através da aplicação sucessiva das funções, onde o domínio é composto por cada um dos valores inteiros contidos entre 0 e o tamanho do painel, e a imagem sendo arredondada para inteiro e deslocada por um valor d (parametrizável), que representa a distância em cada interface. Os demais parâmetros, isto é, α, β, θ, γ, η, ν e ξ são selecionados de forma aleatória uniformemente distribuído em um intervalo definido empiricamente[3].

Utilizando essa estrutura, onde cada ponto é representado por um valor de uma distribuição uniforme definida entre $[-1, 1]$ (conforme exemplo na Figura 5.5a), adicionamos falha neste painel baseando-se no trabalho de Wu et al. [74] (Figura 5.5b), convolvemos com a Ricker e geramos os dados sem ruído (Figura 5.5e).

Para gerar dados com ruído, inserimos alguns passos adicionais às estruturas: *confettis* com tamanho entre 3 e 7 pontos, onde cada região de 256×256 pontos foram adicionados de 250 a 500 *confettis* e ruído gaussiano (Figura 5.5c). Após os ruídos, convolvemos esse painel com uma PSF (escolhida aleatoriamente em nosso conjunto) e geramos os dados ruidosos, conforme Figura 5.5d. Mais amostras são apresentadas na Figura 5.6.

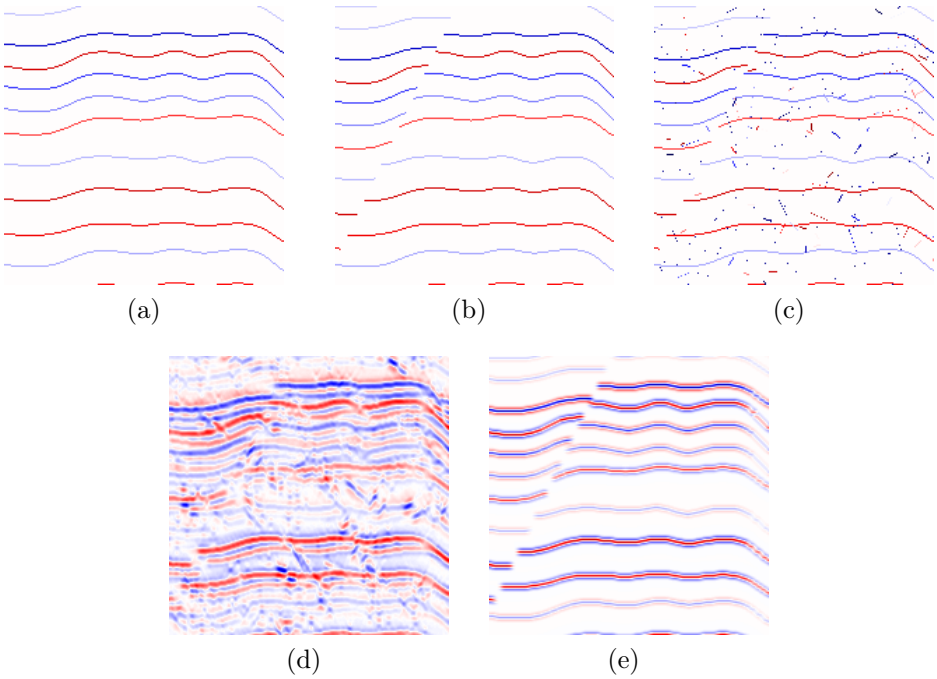
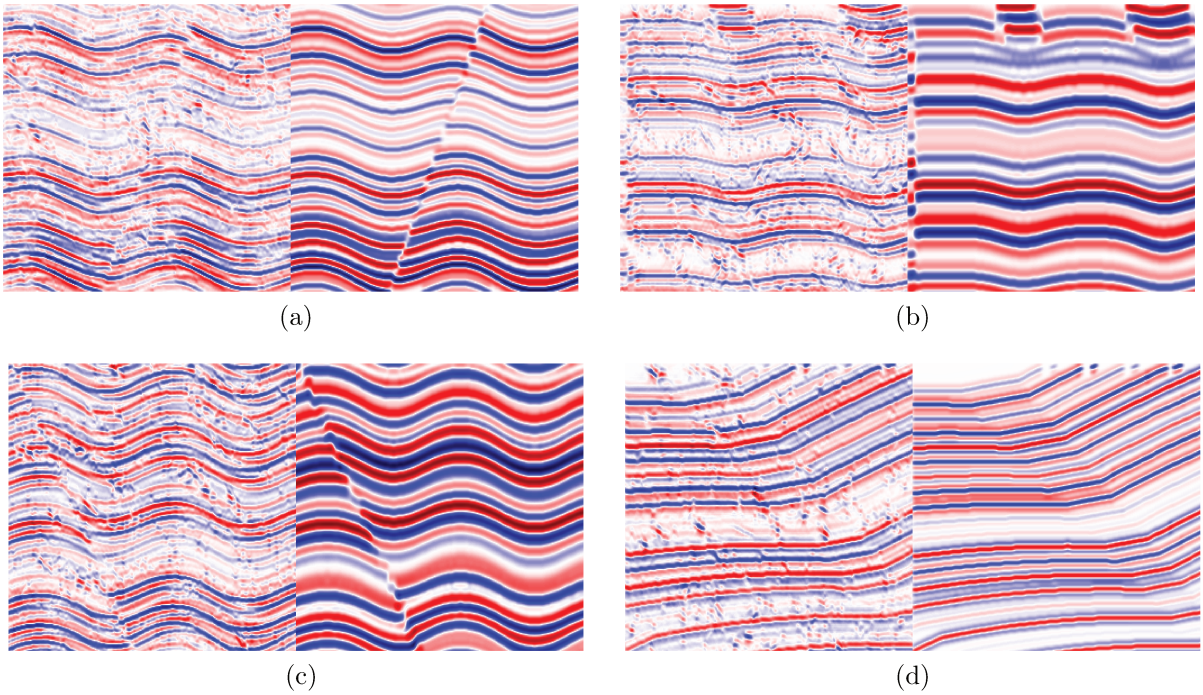


Figura 5.5: Passo a passo para a criação de dados sintéticos com o GD1 e frequência de 50 Hz.

O GD1 é capaz de gerar os dados utilizando todas as CPUs (*Central Processing Unit*, ou Unidade Central de Processamento) disponíveis na máquina e com tamanho de saída menor ou igual a 1024, tanto para dados separados quanto para alinhados (algumas das redes utilizadas recebem como entrada um dado alinhado, isto é, o dado ruidoso e o dado sem ruído são concatenados, gerando assim um dado único). Além do tipo de dados,

[3] $\alpha = [-0.3, 0.3]$, $\beta = [-3, 3]$, $\theta = [0.008, 0.05]$, $\gamma = [1, 5]$, $\eta = [11, 20]$, $\nu = [-5, 5]$ e $\xi = [-5, 5]$.



(a) (b) (c) (d)

Figura 5.6: Exemplos produzidos pelo GD1. Em cada imagem, temos os dados ruidosos à esquerda e os dados sem ruído à direita. Os dados sem ruído foram gerados com frequência aleatória entre 15 e 25 Hz.

podemos configurar também parâmetros como frequência e quantidade de dados. Com isso, geramos 12.000 amostras (10.000 para treinamento e 2.000 para validação). Na Tabela 5.1, apresentamos um resumo da geração de dados paralelizada.

Tabela 5.1: Informações referentes à geração da base de dados com o GD1. Foram geradas 12.000 amostras sintéticas (10.000 para treinamento e 2.000 para validação) distribuídas entre as CPUs disponíveis, permitindo o aumento do desempenho de acordo com a máquina.

CPUs	**Tempo Gasto**	**Tamanho do Painel**	**Frequência**
8	∼17 min	256×256	[15-25] Hz
8	∼1,3 h	512×512	[15-25] Hz
32	∼15 min	512×512	50 Hz

5.1.4 Gerador de Dados – Versão 2 (GD2)

Para o GD2, geramos estruturas planas (Figura 5.7a) e sem espaço entre as camadas refletoras, com valores de $[-1, 1]$. Em seguida, foi adicionada uma região de sal de acordo com a probabilidade $p1 = 30\%$, ou seja, o sal está presente em aproximadamente 30% dos dados gerados (parâmetro configurável), ver Figura 5.7b. Após a adição do sal, realizamos uma deformação por dobramento (Figura 5.7c), por cisalhamento (Figura 5.7d) e por falha (do mesmo modo que na primeira versão, Figura 5.7e), com a probabilidade $p2 = 50\%$, ou seja, a falha está presente em aproximadamente 50% dos dados gerados (parâmetro

configurável). Após estes passos, a estrutura foi convolvida com a Ricker e, assim, geramos os dados sem ruído (Figura 5.7f).

Para a geração dos dados ruidosos, realizamos os passos seguintes: (i) geramos um painel com o mesmo tamanho do painel original que foi preenchido com valores provindos de uma distribuição normal (Gaussiana) centralizada em 0; (ii) aplicamos um filtro gaussiano [30] nesse painel com sigma aleatoriamente escolhido no intervalo $[1.5, 1.8]$; (iii) somamos o painel original com o painel de ruído considerando a relação sinal/ruído desejada (na prática isso define o valor máximo permitido para o ruído, pois o dado ainda deve manter suas características); (iv) adicionalmente, ponderamos quanto ruído será inserido nas região com sal. O resultado final deste processo é apresentado na Figura 5.7g. Finalmente, realizamos a convolução com a PSF (aleatoriamente escolhida dentro do nosso conjunto de PSFs) e geramos o dado ruidoso, conforme Figura 5.7h.

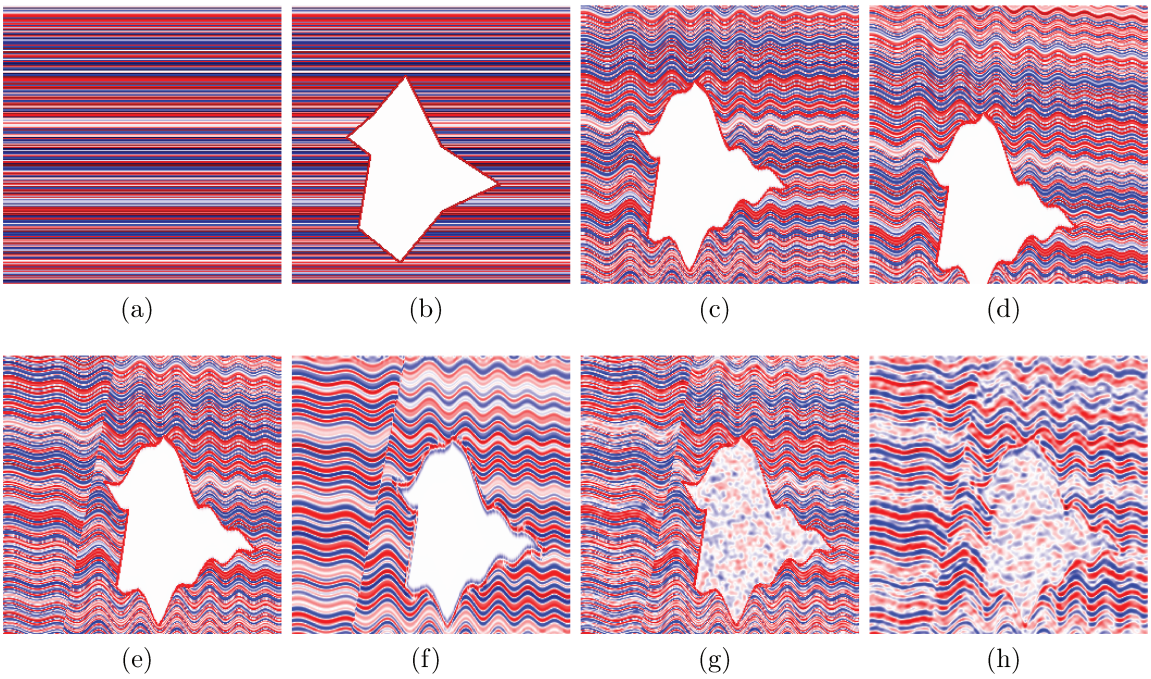


Figura 5.7: Passo a passo para a criação de dados sintéticos com o GD2.

5.2 GANs

Nesta seção, apresentamos os experimentos com as GANs Pix2Pix, Pix2PixHD e SPADE. Para todos os experimentos, utilizamos o GD1.

5.2.1 Pix2Pix

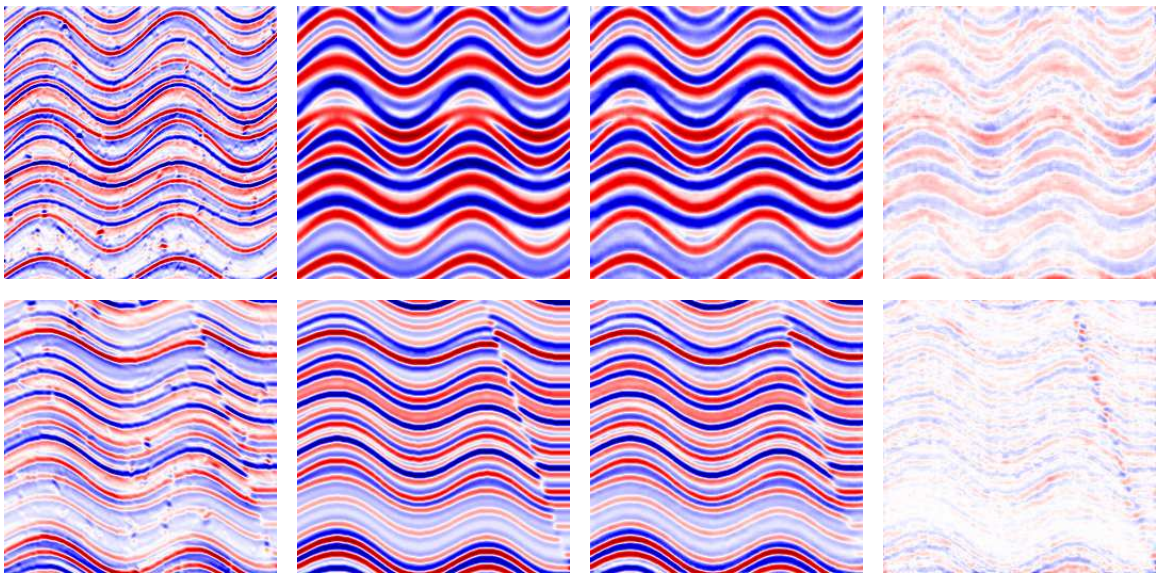
Para treinar nossos modelos baseados na Pix2Pix, utilizamos 10.000 pares de dados sísmicos, uma taxa de aprendizado de 0,002, e um *batch* de tamanho 128. Para 300 épocas, o processo de treinamento levou cerca de 7 horas[4] em uma GPU NVIDIA Tesla V100 com

[4] A análise do treinamento para definir por quanto tempo/épocas a rede deveria ser treinada foi avaliada para cada experimento, considerando fatores como custo, acurácia, potencial de melhoria, etc.

16 GB, alcançando 75, 17% de acurácia no conjunto de dados de validação, ou seja, um conjunto que não foi utilizando na fase de treinamento, composto de 2.000 amostras de dados sísmicos. Alguns resultados são mostrados na Figura 5.8.

Ressaltamos que a entrada da Pix2Pix é uma imagem, logo, os dados sísmicos foram convertidos para imagens antes do treinamento. Com isso, geramos uma base de dados com 10.000 imagens em tons de cinza com a qual foi realizada o treinamento. Do mesmo modo, a saída da rede é equivalente à entrada, então tivemos como saída uma imagem em tons de cinza que foi convertida novamente para dados sísmicos.

Na Figura 5.9 apresentamos o resíduo e diferença SSIM para o exemplo da segunda linha da Figura 5.8. Esse dado apresentou uma melhoria na relação sinal ruído de aproximadamente 26×, e um SSIM entre a imagem alvo e a imagem gerada pela rede de 0,95. Na Figura 5.10 apresentamos o espetro de amplitude para o mesmo, onde é possível perceber que a frequência do dado gerado é equivalente ao do dado sem ruído. Esse modelo será denominado daqui por diante como sendo **Modelo 1**[5], sendo o nosso *baseline*.



(a) (b) (c) (d)

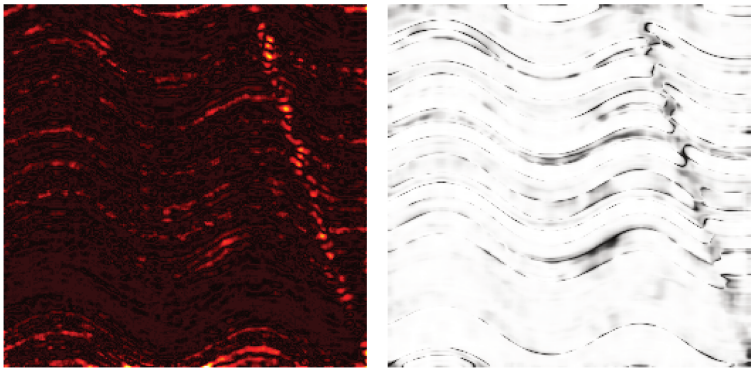
Figura 5.8: Resultados com a Pix2Pix: (a) são as entradas da Pix2Pix, (b) são os dados sem ruído, (c) são as saídas da rede e (d) são as diferenças entre a saída e o dado sem ruído.

5.2.2 Pix2PixHD

Nesta seção, apresentamos os experimentos realizados com a Pix2PixHD, tanto para imagens quanto para dados sísmicos.

Adicionalmente, todos os experimentos reportados neste capítulo obtiveram uma acurácia de treinamento com variação de ± 2% em relação as acurácias de validação reportadas.

[5]Em *Machine Learning*, um modelo de aprendizado de máquina é um arquivo que contém os parâmetros aprendidos durante o processo de treinamento e é utilizado para realizar a predição de novos dados.



(a) (b)

Figura 5.9: Erro no dado gerado pela Pix2Pix para o exemplo da segunda linha da Figura 5.8: (a) é o resíduo e (b) é a diferença SSIM, ambos entre o dado sem ruído e o dado gerado.

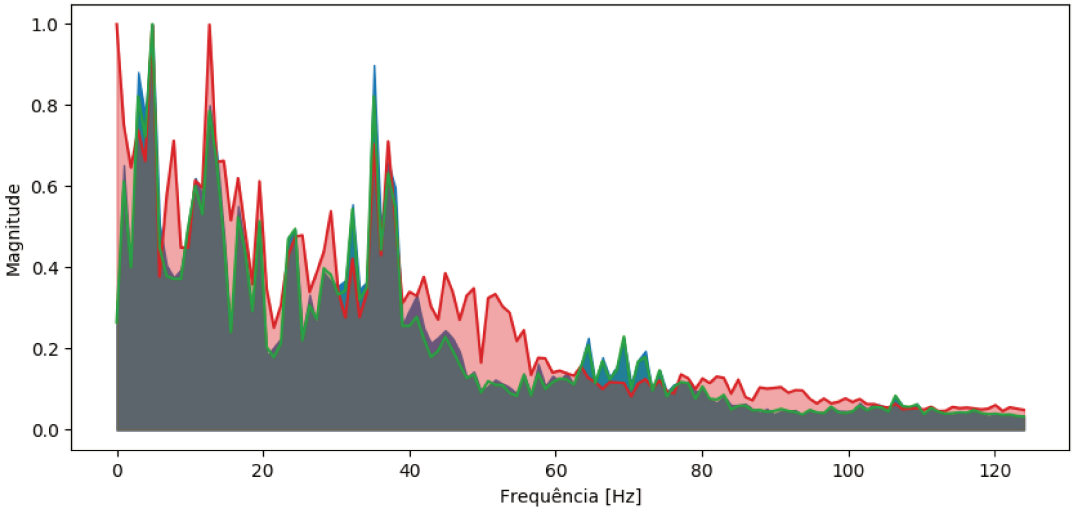


Figura 5.10: Espectro de amplitude do resultado da segunda linha da Figura 5.8 para a Pix2Pix. As curvas azul, vermelho e verde mostram os espectros de Fourier de destino (Y), ruído-entrada – (X) e geração-saída – ($G(X)$) dos dados, respectivamente.

Pix2PixHD com Imagens

Para a Pix2PixHD, reproduzimos o mesmo experimento apresentado na seção anterior (isto é, com o mesmo conjunto de dados) com as mesmas configurações, exceto o tamanho do *batch* igual a 12 (aproximadamente 10× menor que o *batch* da Pix2Pix). Isso ocorre devido à rede Pix2PixHD ter mais parâmetros que a Pix2Pix. Em contrapartida, não é necessário treinar a Pix2PixHD por tantas épocas/iterações quanto a Pix2Pix. Para 143 épocas em uma GPU NVIDIA Tesla V100 com 16 GB, o tempo gasto neste experimento foi de aproximadamente 3 dias (1.651 segundos por época), ou seja, 4× superior ao tempo de treinamento da Pix2Pix.

O resultado sobre o mesmo conjunto de validação foi de $80,82\%$. Essa acurácia é superior a Pix2Pix em aproximadamente cinco pontos percentuais. Dado esse resultado, decidimos continuar com os nossos experimentos apenas com a Pix2PixHD. Vale a

pena ressaltar que, conforme mostrado, o custo computacional para o treinamento com a Pix2PixHD é superior ao treinamento da Pix2Pix, sendo a melhora nos resultados a justificativa para a troca das arquiteturas. Esse modelo foi denominado como **Modelo 2**.

Dando continuidade aos experimentos, aumentamos a dimensão dos dados, visto que essa rede foi projetada para gerar dados em alta resolução. Geramos uma nova base de dados com as mesmas características e parâmetros da anterior, porém com uma resolução de 512×512 ao invés de 256×256. O treinamento levou 2.656 segundos por época, o que representa 1,6× o tempo de treinamento com a resolução anterior. Quanto aos resultados, obtivemos $84,05\%$ de acurácia, um aumento de quatro pontos percentuais em relação ao modelo anterior, o que também nos forneceu suporte para a mudança do tamanho do dado de entrada. Vale ressaltar que não realizamos experimentos com resoluções ainda maiores devido ao aumento da dificuldade e do tempo de treinamento, visto que quanto maior os dados, maior a quantidade de parâmetros a ser aprendido e, consequentemente, mais recursos computacionais serão necessários. Esse experimento gerou nosso **Modelo 3**.

Pix2PixHD com Dados Sísmicos

Neste experimento, adicionamos o treinamento direto com os dados sísmicos ao invés de imagens e também o ganho de altas frequências como objetivo à rede por meio da geração de dados sem ruído com alta frequência, visando à conversão de dados de baixas frequências para altas frequências. Para isso, geramos também uma nova base de dados com a frequência da Ricker configurada para 50 Hz, para a mesma quantidade de dados.

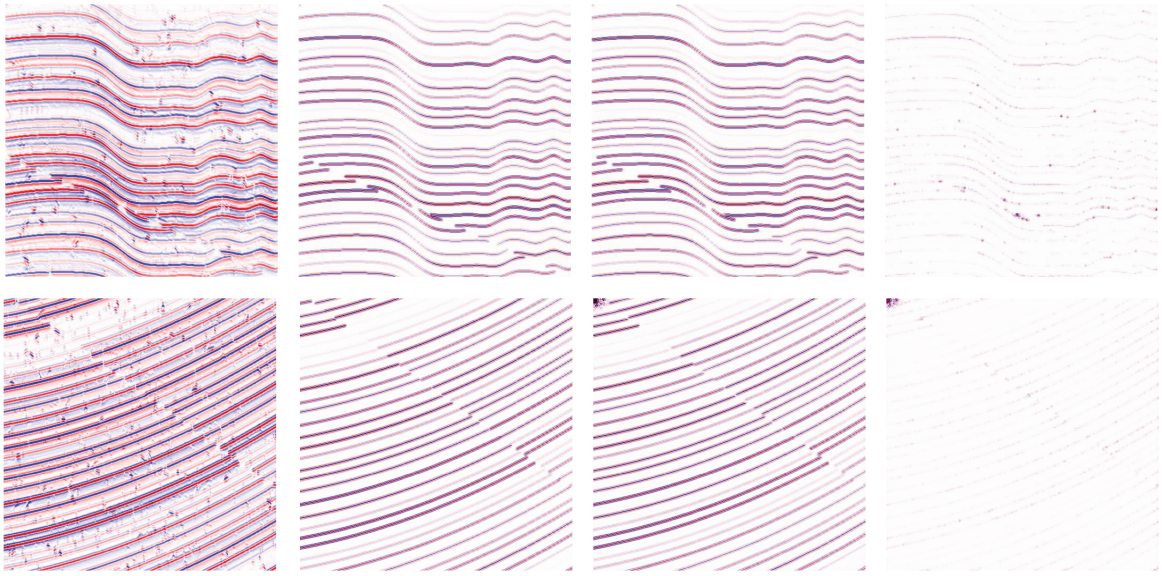
Treinamos a Pix2PixHD com *batch* de tamanho 24 por 173 épocas em quatro GPUs NVIDIA Tesla V100 com 16 GB (totalizando 64 GB de memória). Cada época demorou 580 segundos, levando um total de aproximadamente 24 horas. Como resultado alcançamos uma acurácia de $82,77\%$, o que é inferior ao experimento anterior, entretanto, justificável devido à adição das altas frequências. Esse experimento gerou o **Modelo 4**.

No experimento seguinte, removemos todas as transformações e normalizações realizadas pela Pix2PixHD. Como essas transformações foram projetadas para imagem, elas poderiam estar prejudicando resultado da rede uma vez que a entrada são dados sísmicos. Treinamos novamente o modelo e obtivemos $86,65\%$ de acurácia, gerando o **Modelo 5**. Na Figura 5.11, apresentamos alguns exemplos de dados gerados com esse modelo.

Na Figura 5.12, apresentamos o resíduo e a diferença SSIM para o exemplo da segunda linha da Figura 5.11. Esse dado apresentou uma melhoria na relação sinal-ruído de aproximadamente 12×, e um SSIM entre a imagem alvo e a imagem gerada pela rede de 0,98. Na Figura 5.13, apresentamos o espectro de amplitude para o mesmo, onde é possível perceber que a frequência do dado gerado é equivalente à frequência do dado sem ruído. Além disso, também é possível visualizar o ganho de altas frequências.

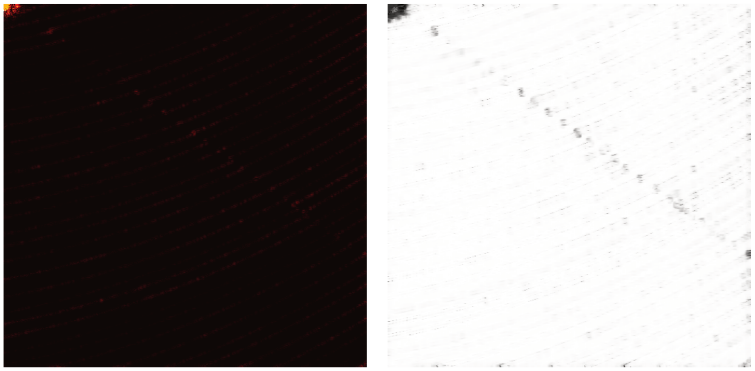
Para o próximo experimento, realizamos a substituição das camadas de *instance normalization* por *batch normalization*[6]. Alcançamos uma acurácia, referente ao conjunto de validação, de $88,49\%$. Entretanto, para alguns dados, foram introduzidos artefatos, o que não condiz com o que estamos buscando, conforme apresentado na Figura 5.14. Nesse

[6]Uma explicação detalhada sobre os tipos de normalização pode ser encontrada em `https://mlexplained.com/2018/11/30/an-overview-of-normalization-methods-in-deep-learning/`



(a) (b) (c) (d)

Figura 5.11: Resultados com a Pix2PixHD: (a) são as entradas da Pix2PixHD, (b) são os dados sem ruído, (c) são as saídas da rede e (d) são as diferenças entre a saída e o dado sem ruído.



(a) (b)

Figura 5.12: Erro no dado gerado pela Pix2PixHD para o exemplo da segunda linha da Figura 5.11: (a) é o resíduo e (b) é a diferença SSIM, ambos entre o dado sem ruído e o dado gerado.

experimento, geramos o **Modelo 6**.

5.2.3 SPADE

Para a SPADE não realizamos nenhum experimento com imagens, visto que os experimentos anteriores demostraram que a utilização de dados sísmicos produz melhores resultados do que imagens. Dito isso, fizemos as modificações necessárias para utilizar a SPADE com dados sísmicos brutos. Também removemos transformações e normalização, assim como na Pix2PixHD.

Para o primeiro experimento com a SPADE, utilizamos duas GPUs Titan X (totalizando 24 GB), um tamanho de *batch* de 2 e treinamos o modelo por 50 épocas. O

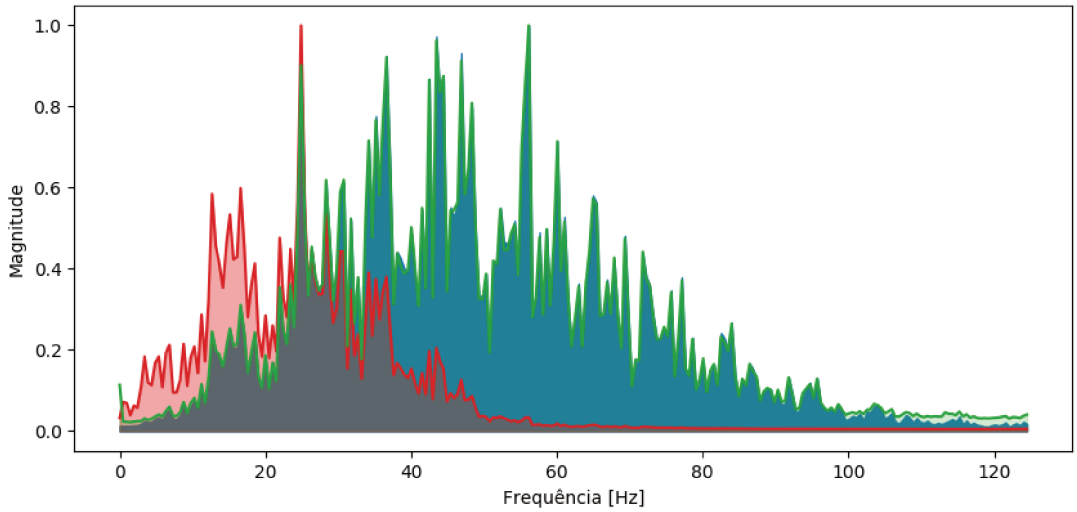
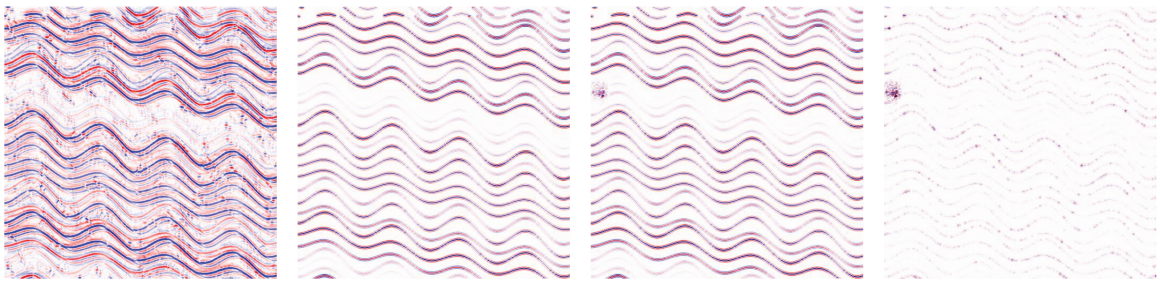


Figura 5.13: Espectro de amplitude do resultado da segunda linha da Figura 5.11 para a Pix2PixHD. As curvas azul, vermelho e verde mostram os espectros de Fourier de destino (Y), ruído-entrada – (X) e geração-saída – ($G(X)$) dos dados, respectivamente.



(a) (b) (c) (d)

Figura 5.14: Artefato introduzido quando utilizando *batch normalization*: (a) é a entrada da Pix2PixHD, (b) é o dado sem ruído, (c) é a saída da rede e (d) é a diferença entre a saída e o dado sem ruído. É possível analisar em (c) — dado gerado — que um artefato foi introduzido. Esse fenômeno acontece em apenas alguns exemplos.

modelo dessa rede tem mais parâmetros do que a Pix2PixHD, mas o número de épocas necessário para o modelo convergir é menor. Cada época levou aproximadamente 5.100 segundos, ou seja, o tempo de total de aproximadamente 3 dias, sendo o mesmo tempo da Pix2PixHD com o Modelo 2, por exemplo, porém utilizando menos memória. Nesse experimento geramos o **Modelo 7** com acurácia de $88,35\%$, equivalente a Pix2PixHD.

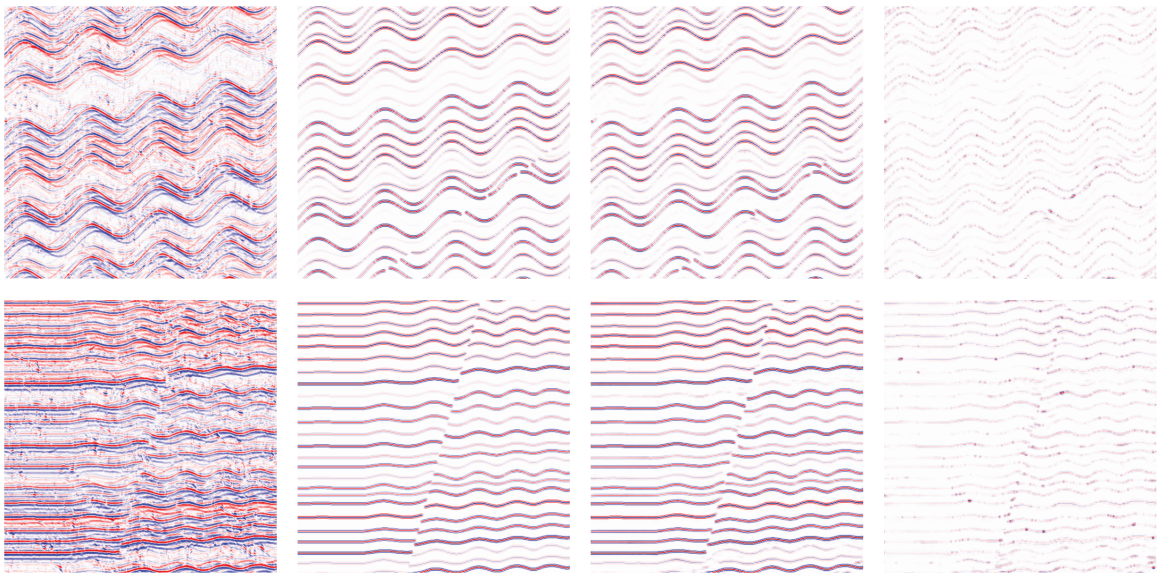
No próximo experimento, adicionamos uma camada de *upsampling* entre cada dois blocos residuais [25] da SPADE (baseado no trabalho de Wang et al. [71]). Essas camadas adicionais aumentaram a quantidade de parâmetros da rede, o que elevou o tempo de treinamento. Treinamos com os mesmos hiperparâmetros do experimento anterior e como resultado conseguimos $87,89\%$ de acurácia, gerando o **Modelo 8**. Ainda seguindo a mesma linha de experimentação (e se baseando no trabalho mencionado anteriormente), adicionamos mais uma camada de *upsampling* e mais um bloco residual no final do gerador. O resultado foi de $86,87\%$, sendo este o nosso pior resultado para SPADE (**Modelo 9**).

Para o **Modelo 10**, utilizamos a arquitetura do Modelo 7 e do Modelo 8, realizando

alguns experimentos referentes à taxa de aprendizado (*learning rate*). O melhor resultado dentre todos esses experimentos não foi superior ao do Modelo 7, pois alcançamos apenas $86,38\%$ de acurácia.

Por fim, realizamos mais dois experimentos com a SPADE utilizando as arquiteturas dos Modelos 9 e 10, por acreditar que a adição das camadas mencionadas anteriormente ainda poderia melhorar os resultados dos experimentos, e devido ao fato de que a adição dessas camadas atenuou os erros nas regiões de bordas. Diante desses motivos, aumentamos o tamanho da entrada em 40 píxeis antes de passar pela rede e descartamos 40 píxeis da saída da rede. Com isso conseguimos melhorar os nossos resultados, sendo $87,08\%$ (para a arquitetura do Modelo 9) e $88,48\%$ (para arquitetura do Modelo 10), gerando os **Modelos 11** e **12**, finalizando assim os experimentos com a SPADE.

Na Figura 5.15, apresentamos alguns exemplos de dados gerados com o Modelo 12, enquanto na Figura 5.16 apresentamos o resíduo e a diferença SSIM para o exemplo da segunda linha da Figura 5.15. Esse dado apresentou uma melhoria na relação sinal-ruído de aproximadamente $7\times$, e um SSIM entre a dado alvo e o dado gerada pela rede de 0,96. Na Figura 5.17, apresentamos o espectro de amplitude para o mesmo, onde é possível perceber que a frequência do dado gerado é equivalente à frequência do dado sem ruído. Além disso, também é possível visualizar o ganho de altas frequências.

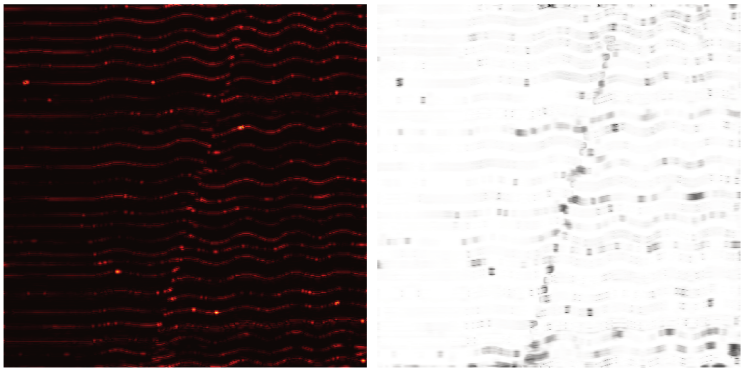


(a) (b) (c) (d)

Figura 5.15: Resultados com a SPADE: (a) são as entradas da SPADE, (b) são os dados sem ruído, (c) são as saídas da rede e (d) são as diferenças entre a saída e o dado sem ruído.

5.2.4 GD1 & GANs: Resumo dos Resultados

Na Tabela 5.2, apresentamos uma comparação entre resultados dos experimentos realizados com as três redes. O Modelo 1 foi treinado com a Pix2Pix, já os Modelos de 2 a 6 foram treinados com a Pix2PixHD, enquanto para os Modelos de 7 até 12 foram treinados com a SPADE. Nesta tabela, além do modelo, também são apresentadas a frequência dos



(a) (b)

Figura 5.16: Erro no dado gerado pela SPADE para o exemplo da segunda linha da Figura 5.15: (a) é o resíduo e (b) é a diferença SSIM, ambos entre o dado sem ruído e o dado gerado.

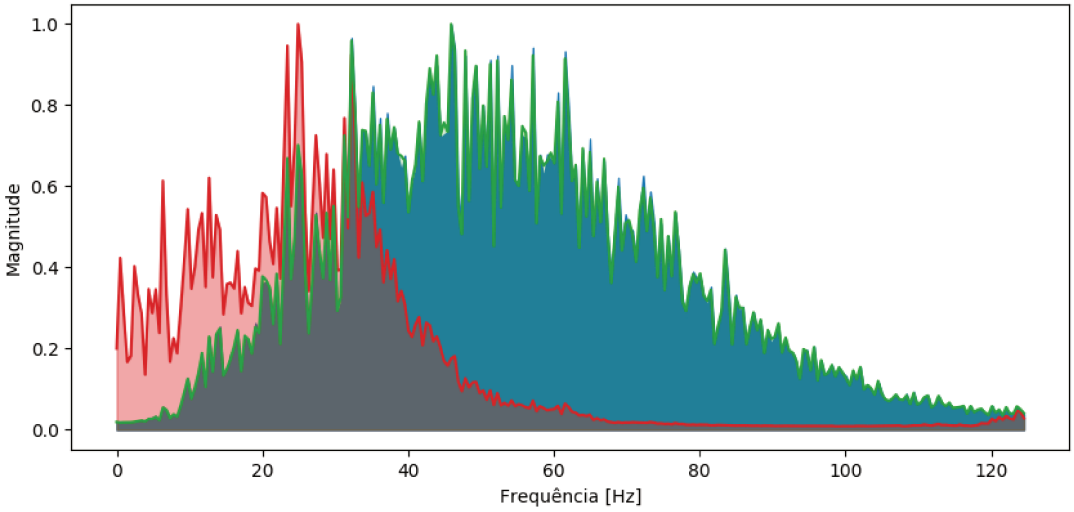


Figura 5.17: Espectro de amplitude do resultado da segunda linha da Figura 5.15 para a SPADE. As curvas azul, vermelho e verde mostram os espectros de Fourier de destino (Y), ruído-entrada – (X) e geração-saída – ($G(X)$) dos dados, respectivamente.

dados sem ruído utilizada, o total de memória de GPU utilizada (maiores detalhes sobre as GPUs são apresentados na Tabela 5.3), o tamanho do *batch*, o número de épocas, o tempo gasto por época e a acurácia de cada modelo.

Com esses resultados concluímos que o melhor modelo da Pix2Pix alcançou $75,17\%$ de acurácia (Modelo 1), o melhor modelo da Pix2PixHD alcançou $86,65\%$ (Modelo 5) (desconsiderando o Modelo 6 que introduziu artefatos nos dados) e o melhor modelo da SPADE alcançou $88,48\%$ de acurácia (Modelo 12). Considerando os resultados e os requisitos necessários para treinar os modelos (por exemplo, número de parâmetros, números de GPUs e memória das GPUs), decidimos que o nosso Modelo 5 da Pix2PixHD será o padrão a ser utilizado nos próximos experimentos com o GD2.

Tabela 5.2: Resumo dos treinamentos com a Pix2Pix, a Pix2PixHD e a SPADE.

Modelo	Frequência (Hz)	Memória (GB)	Batch	Épocas	Tempo por Época (s)	Acurácia (%)
1	[15, 25]	16	128	267	∼93	**75,17**
2	[15, 25]	16	12	143	∼1.651	80,82
3	[15, 25]	16	6	143	∼2.656	84,05
4	50	64	24	173	∼580	82,77
5	50	128	48	195	∼356	**86,65**
6	50	22	8	200	∼2.337	88,49
7	50	24	2	50	∼5.100	88,35
8	50	32	4	50	∼4.200	87,89
9	50	48	9	150	∼2.000	86,87
10	50	48	9	150	∼2.000	86,38
11	50	32	6	150	∼2.669	87,08
12	50	32	6	150	∼2.669	**88,48**

Tabela 5.3: Configuração de GPUs utilizadas para gerar o total de memória utilizada.

Modelo	Memória (GB)	Quantidade	Memória Total (GB)
V100	16	1	16
V100	16	4	64
V100	16	8	128
GTX 1080	11	2	22
Titan X	12	2	24
RTX 5000	16	3	48
RTX 5000	16	2	32

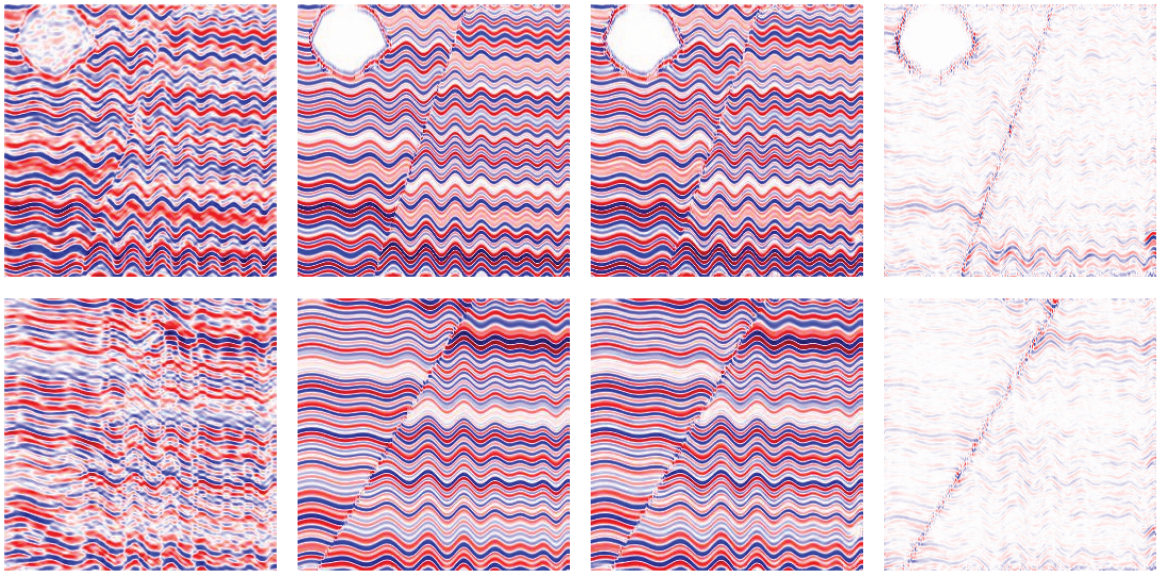
5.3 GD2 & Pix2PixHD

Conforme apresentado na Seção 5.1.4, os dados produzidos com o GD2 são mais difíceis de serem treinados, tornando a tarefa de remoção de ruídos mais desafiadora.

Para o treinamento, utilizamos a mesma configuração do Modelo 5 com 10.000 dados de treinamento e 2.000 de validação. Ressaltamos que os dados gerados com o GD2 não têm ganho de frequência. Esse treinamento foi realizado por 50 épocas alcançando um resultado de $75,44\%$ e gerou o **Modelo 13**.

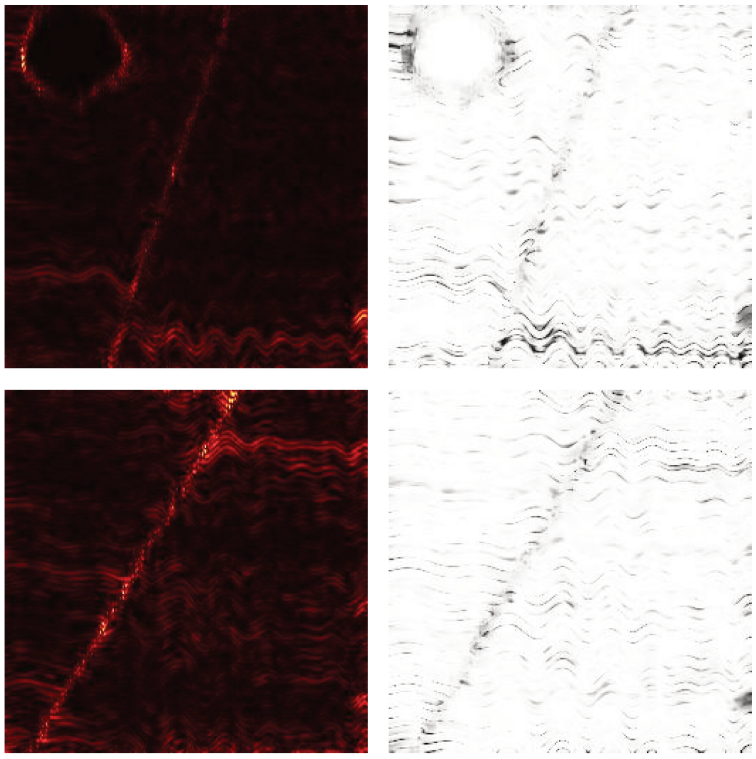
Na Figura 5.18, apresentamos dois exemplos de dados gerados com o Modelo 13, enquanto na Figura 5.19 apresentamos o resíduo e a diferença SSIM para os exemplos da Figura 5.18. Para o exemplo da primeira linha, tivemos uma melhoria na relação sinal-ruído de aproximadamente 3×, e SSIM entre o dado gerado e o dado sem ruído de 0,94. Para o exemplo da segunda linha tivemos uma melhoria na relação sinal-ruído de aproximadamente 6× e SSIM entre o dado gerado e o dado sem ruído de 0,96. Na Figura 5.20 apresentamos o espectro de amplitude para esses exemplos.

Na próxima seção, avaliamos diferentes funções de custo com o objetivo de melhorar



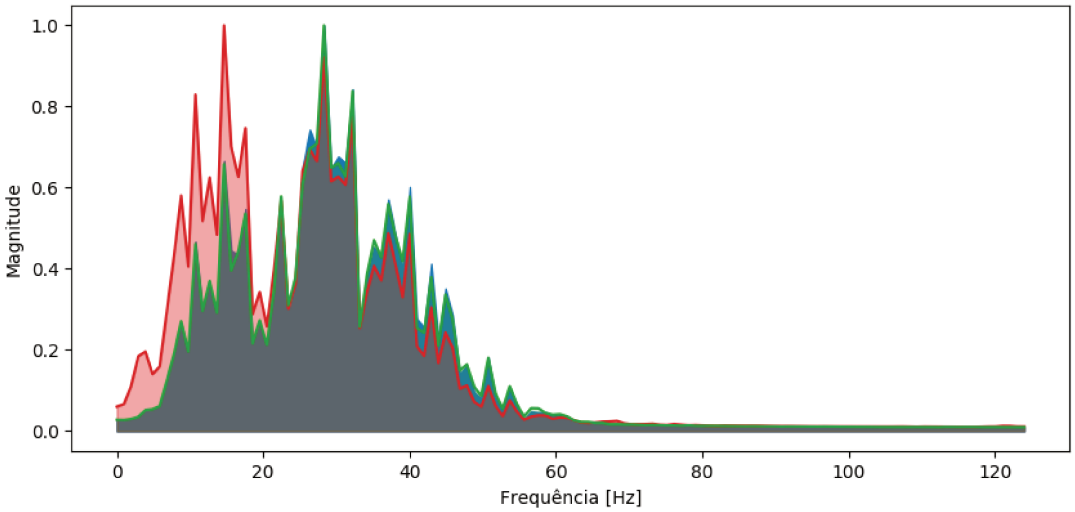
(a) (b) (c) (d)

Figura 5.18: Resultados com os dados gerados pelo GD2 e treinados pela Pix2PixHD: (a) são os dados com ruídos, (b) são os dados sem ruído, (c) são os dados gerado pela rede e (d) são as diferenças entre a saída e o dado sem ruído.

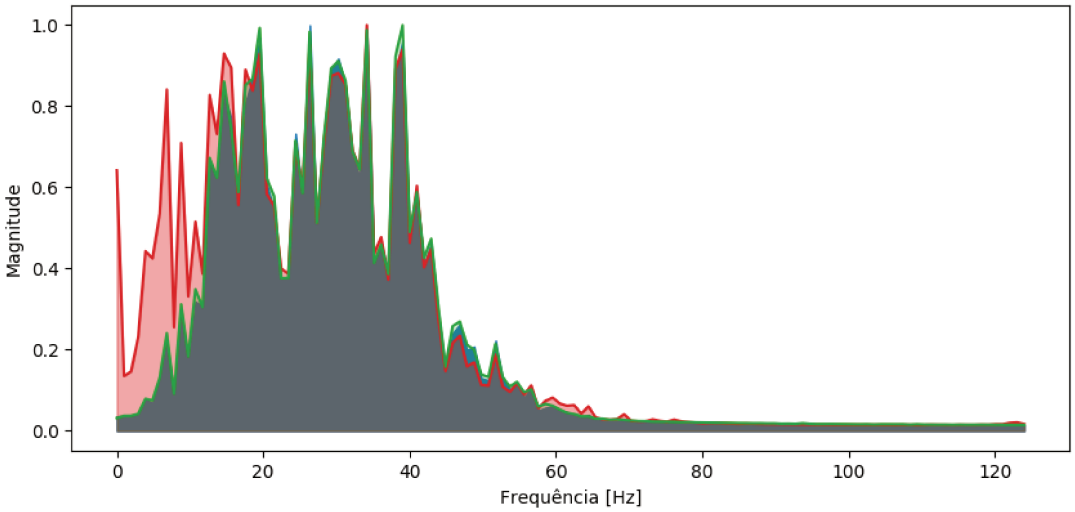


(a) (b)

Figura 5.19: Erros nos dados gerados pela Pix2PixHD para os exemplos da Figura 5.18: (a) representam o resíduo, (b) representam a diferença SSIM, ambos entre o dado sem ruído e o dado gerado.



(a)



(b)

Figura 5.20: Espectro de amplitude do resultado da Figura 5.18 para a Pix2PixHD. As curvas azul, vermelho e verde mostram os espectros de Fourier de destino (Y), ruído-entrada – (X) e geração-saída – ($G(X)$) dos dados, respectivamente.

os resultados. Para tanto, consideramos o Modelo 13 como sendo nosso *baseline*.

5.4 Avaliação de Funções de Custo

O primeiro passo nos experimentos referente à função de custo é a verificação da importância que cada função de custo nativa da Pix2PixHD gera nos resultados. Para isso, removemos uma função de custo por vez. Conforme apresentado na Seção 2.1.2, a Pix2PixHD tem três funções de custo: GAN ($\mathcal{F}_{\text{HD}}$, Equação 2.8), *Feat* ($\mathcal{F}_{\text{FM}}$, Equação 2.5, e *perceptual* ($\mathcal{F}_{\text{P}}$, Equação 2.6). A combinação dessas funções funções gera uma função de custo final conforme Equação 2.7. Relembramos a equação aqui para facilitar a compreensão.

$$\begin{aligned}\mathcal{L}_{\text{HD}}(G_*, \{D_k\}_*) &= \min_G \left(\left(\max_{D_1,D_2,D_3} \sum_{k=1,2,3} \mathcal{F}_{\text{HD}}\left(G, D_k\right) \right) \right. \\ &+ \left. \lambda_1 \sum_{k=1,2,3} \mathcal{F}_{\text{FM}}\left(G, D_k\right) + \lambda_2 \mathcal{F}_{\text{P}}(G) \right).\end{aligned}$$

A função de custo da GAN ($\mathcal{F}_{\text{HD}}$, Equação 2.8) não pode ser removida, pois ela é responsável pela metodologia de jogo contraditório de treinamento das GANs. Dito isso, a primeira função que removemos foi a *Feat*, que apresentou uma variação sútil na acurácia passando de $75,44\%$ para $74,86\%$ (**Modelo 14**), indicando que a função tem um impacto pequeno sobre o treinamento. Entretanto, teve um impacto significativo sobre a qualidade visual dos resultados; logo, optamos por mantê-la.

Na próxima seção, repetimos o experimento com a função de custo *Perceptual*. Com ela, houve uma diferença significativa na acurácia, de $75,44\%$ para $71,21\%$ (**Modelo 15**), logo, também a mantivemos. Com isso, mantivemos todas as funções de custo nativas e acrescentamos as funções de custo que apresentaremos a seguir. O resumo dessa remoção de função de custo pode ser visto na Tabela 5.4.

Tabela 5.4: Resultados para a remoção de funções de custo da Pix2PixHD.

Modelo	**Função Removida**	**Acurácia (%)**
13	–	75,44
14	*Feat*	74,86
15	*Perceptual*	71,21

Em seguida, também testamos as novas funções de custo implementadas, que é definida por $\mathcal{F}_{\text{loss}}$ na Equação 4.4 com peso λ_3. Relembramos a equação aqui para facilitar a compreensão.

$$\begin{aligned}\mathcal{L}_{\text{cost}}(G_*, \{D_k\}_*) &= \min_G \left(\left(\max_{D_1,D_2,D_3} \sum_{k=1,2,3} \mathcal{F}_{\text{HD}}\left(G, D_k\right) \right) \right. \\ &+ \left. \lambda_1 \sum_{k=1,2,3} \mathcal{F}_{\text{FM}}\left(G, D_k\right) + \lambda_2 \mathcal{F}_{\text{P}}(G) + \lambda_3 \mathcal{F}_{\text{loss}}\left(G\right) \right),\end{aligned}$$

onde $\mathcal{F}_{\text{loss}}$ será definida de acordo com as medidas a seguir.

A primeira função de custo que desenvolvemos, a $\mathcal{F}_{\text{SEISMIC}}$ (Equação 4.6), foi baseada na fórmula de acurácia e foi desenvolvida com o auxílio dos especialistas em sísmica do HPG/CEPETRO. Adicionamos essa função de custo às outras da Pix2PixHD. Inicialmente, ponderamos essa função de custo com peso um (isto é, $\lambda_3 = 1$), ou seja, o valor mínimo dessa função de custo é 0 e o máximo é 1. Obtivemos o resultado de $75,81\%$ (**Modelo 16**), o que não é relevante em relação ao anterior. No próximo experimento, aumentamos o peso dessa função de custo para 10 (isto é, $\lambda_3 = 10$), sendo mais compatível com as outras funções de custos. Com isso, obtivemos um resultado de $77,12\%$ (**Modelo 17**) o que é significativamente melhor em relação ao resultado utilizando somente as funções de custo da Pix2PixHD.

Em seguida, implementamos uma função baseada na SSIM ($\mathcal{F}_{\text{SSIM}}$, Equação 4.8). Adicionamos essa função de custo com peso 1 e 10 ($\lambda_3 = 1$ e $\lambda_3 = 10$), obtendo $76,18\%$ (**Modelo 18**) e $76,56\%$ (**Modelo 19**) de acurácia, respectivamente. Visto que tivemos uma melhoria (mesmo que pequena) ao utilizar essa função de custo, então trabalhamos em mais duas variações da mesma. Nas duas implementações anteriores, a função de custo $\mathcal{F}_{\text{SSIM}}$ são forçadas a retornar valores maiores ou igual a zero, isso é realizado para evitar valores inconsistentes no cálculo. Na nova versão, a função pode retornar valores negativos (assim como a SSIM original), entretanto, essa função de custo só é ativada quando os dados gerados atingem um valor de SSIM maior que zero. O resultado desse experimento foi $75,91\%$ (**Modelo 20**), sendo este inferior aos resultados anteriores.

Para o próximo experimento, utilizamos uma função de custo baseada na MS-SSIM ($\mathcal{F}_{\text{MSSSIM}}$, Equação 4.9). De modo geral, seguimos as mesmas regras dos experimentos com as funções de custos $\mathcal{F}_{\text{SSIM}}$. Os resultados para ponderação 1 e 10 ($\lambda_3 = 1$ e $\lambda_3 = 10$) foram de $75,56\%$ e $77,19\%$ (**Modelos 21** e **22**), respectivamente. Percebemos que o treinamento com a função de custo $\mathcal{F}_{\text{MSSSIM}}$ com peso 10 ($\lambda_3 = 10$) apresentou uma melhoria relevante em relação ao *baseline*, finalizando assim nossos experimentos com funções de custo baseada em SSIM e MS-SSIM.

A nossa última função de custo desenvolvida foi a *frequency loss* ($\mathcal{F}_{\text{FREQ}}$, Equação 4.10), onde a função SIM utilizada foi a mesma função de perda de conteúdo utilizada na Pix2Pix, isto é, $\mathcal{F}_{\text{lost}}$ (Equação 2.2). Os resultados em termos de acurácia não foram significativamente superiores, acalçando $76,08\%$ (**Modelo 23**), entretanto, notamos um melhoria quanto ao ganho de frequência. Para finalizar os experimentos com funções de custos, testamos a função de custo Normal ($\mathcal{F}_{\text{NORMAL}}$, Equação 4.13) apresentada por Geng et al. [22], e percebemos uma sutil melhora na qualidade visual dos dados gerados, apesar da acurácia não apresentar melhorias, com $75,92\%$ (**Modelo 24**). O resumo de todos esses resultados são apresentados na Tabela 5.5.

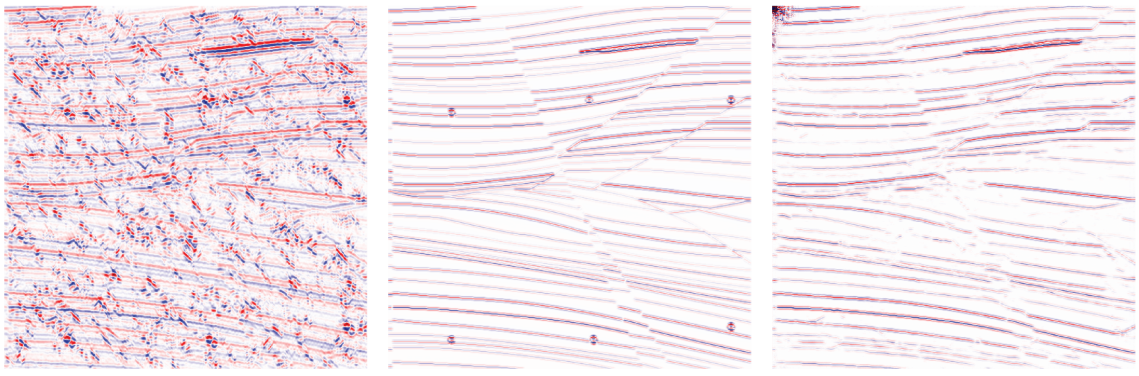
Tabela 5.5: Resumo dos experimentos com funções de custo.

Modelo	**Função Adicionada**	**Peso (λ_3)**	**Acurácia (%)**
13	-	–	75,44
16	$\mathcal{F}_{\text{SEISMIC}}$	1	75,81
17	$\mathcal{F}_{\text{SEISMIC}}$	10	77,12
18	$\mathcal{F}_{\text{SSIM}}$	1	76,18
19	$\mathcal{F}_{\text{SSIM}}$	10	76,56
20	$\mathcal{F}_{\text{SSIM}}$	10	75,91
21	$\mathcal{F}_{\text{MSSSIM}}$	1	75,56
22	$\mathcal{F}_{\text{MSSSIM}}$	10	**77,19**
23	$\mathcal{F}_{\text{FREQ}}$	1	76,08
24	$\mathcal{F}_{\text{NORMAL}}$	1	75,92

5.5 Validação no *Sigsbee* e Dados Reais

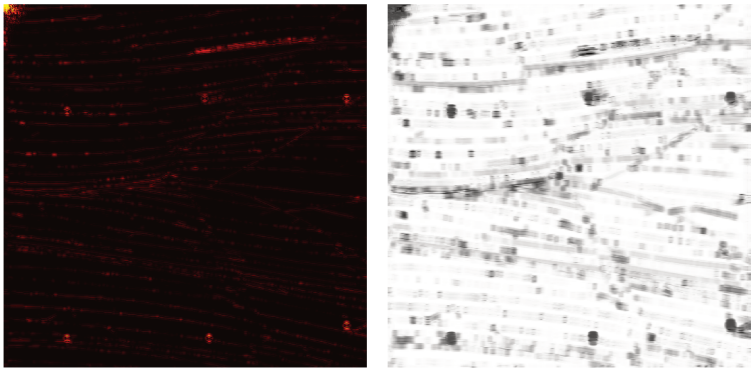
Os resultados apresentados até o momento referem-se apenas ao nosso próprio conjunto de dados. Nesta seção, apresentamos o resultado no *Sigsbee2a*, um dado sintético conhecido na literatura disponibilizado pelo consórcio SMAART JV, e em alguns dados reais disponibilizados pelo HPG/CEPETRO. Para tal, selecionamos dois modelos: o Modelo 5 (o melhor com o GD1) e o Modelo 22 (o melhor com o GD2).

Ao aplicar o Modelo 5 no *Sigsbee2a* [12] — um dado sinético disponível publicamente e bastante utilizado na literatura — conseguimos uma acurácia de $75,51\%$, o que demonstra que nossos modelos podem ser utilizados em dados que não foram produzidos por nossos geradores. Na Figura 5.21, apresentamos um exemplo que corresponde a uma região 512×512 desse dado. Apesar do bom resultado, o dado de entrada da rede possuía alguns difratores que foram removidos incorretamente nos dados gerados. Na Figura 5.22, apresentamos os resíduos e a diferença SSIM, enquanto na Figura 5.23 apresentamos a amplitude de frequência, onde é possível perceber o ganho de frequência.



(a) (b) (c)

Figura 5.21: Resultados com o *Sigsbee2a*: (a) é o dado ruidoso, (b) é a dado sem ruído, (c) é o dado gerado.



(a) (b)

Figura 5.22: Erro no dado gerado da Figura 5.21 no *Sigsbee*: (a) é o resíduo e (b) é a diferença SSIM, ambos entre o dado sem ruído e o dado gerado.

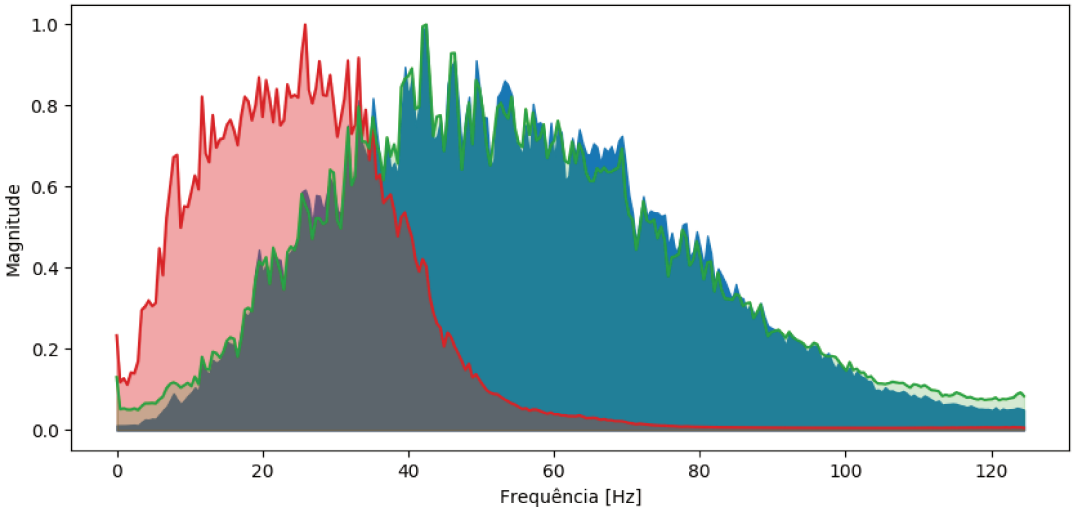
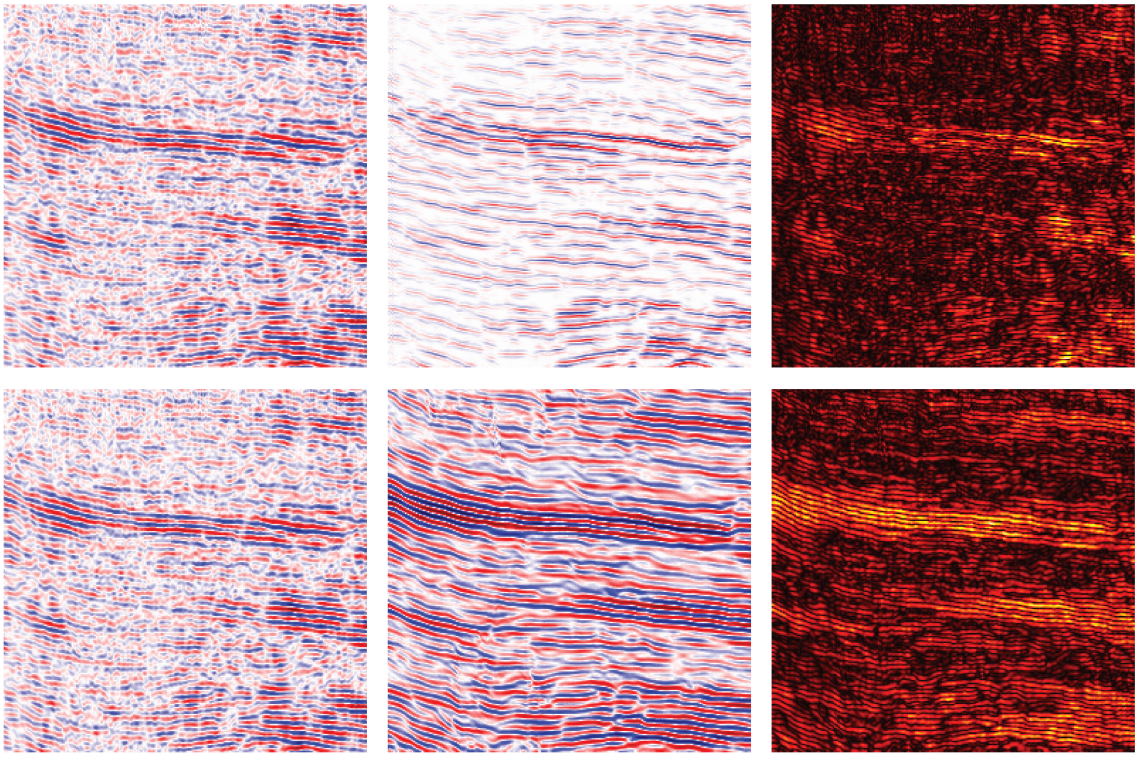


Figura 5.23: Espectro de amplitude do resultado do *Sigsbee* da Figura 5.21. As curvas azul, vermelho e verde mostram os espectros de Fourier de destino (Y), ruído-entrada – (X) e geração-saída – ($G(X)$) dos dados, respectivamente.

Para dados reais, utilizamos ambos os mesmos modelos (5 e 22) para demostrar a diferença entre os modelos treinados com diferentes bases de dados e com ganho de frequência (Modelo 5) e sem ganho de frequência (Modelo 22). Apresentamos três amostras reais, todas da região de Tacutu (Roraima, Brasil), com tamanhos de 512×512. Nas Figuras 5.24, 5.26 e 5.28, apresentamos os exemplos 1, 2 e 3, respectivamente. Na primeira linha apresentamos o Modelo 5 e na segunda linha o Modelo 22, sendo que (a) e (d) correspondem ao dado real original, (b) e (e) ao dado gerado, e (c) e (f) ao resíduo entre o dado gerado e o dado original, ou seja, podemos visualizar o que foi removido ou modificado do dado original. Nas Figuras 5.25, 5.27 e 5.29, apresentamos os espectros de amplitude referentes aos exemplos 1, 2 e 3, respectivamente.

Os resultados apresentados nesta seção foram avaliados por especialistas em sísmica, que reportaram que os resultados para remoção de ruídos em dados sintéticos (aproximadamente 80% de acurácia) estão bons o suficiente para testar em dados reais. Em relação



(a) (b) (c)

Figura 5.24: Resultados da predição em dados reais – Exemplo 1: (a) é o dado real, (b) é o dado gerado e (c) é a diferença entre o real e o dado gerado. Na primeira linha, apresentamos os resultados utilizando o Modelo 5 e na segunda linha o Modelo 22.

às análises de dados reais, os especialistas reportaram que os resultados também foram interessantes. Entretanto, ressaltaram que o dado real utilizado é terrestre, ou seja, um dado complexo de ser analisado quando comparado com um dado marítimo. Além disso, para dado real não se tem uma verdade absoluta para fazer uma comparação mais precisa. Assim, a análise em dados reais deve ser realizada por um especialista que, preferencialmente, tenha um conhecimento prévio do dado (ou da região onde foi feita a aquisição). Outro ponto importante mencionado é que, apesar da dificuldade de analisar a remoção de ruídos, o ganho de frequência é evidente e pode ser avaliado com mais detalhes, e que o ganho de altas frequências trouxe melhorias para o dado.

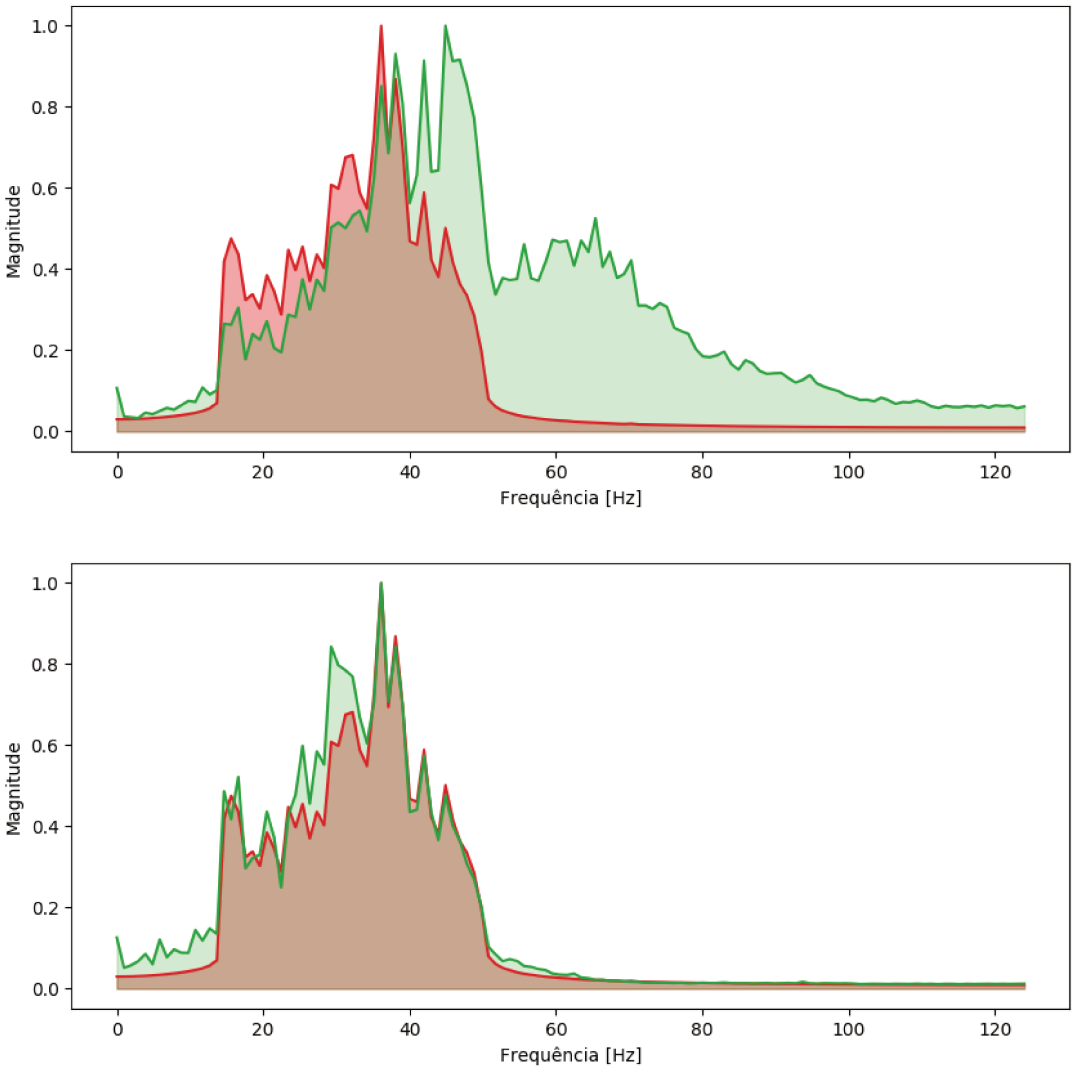
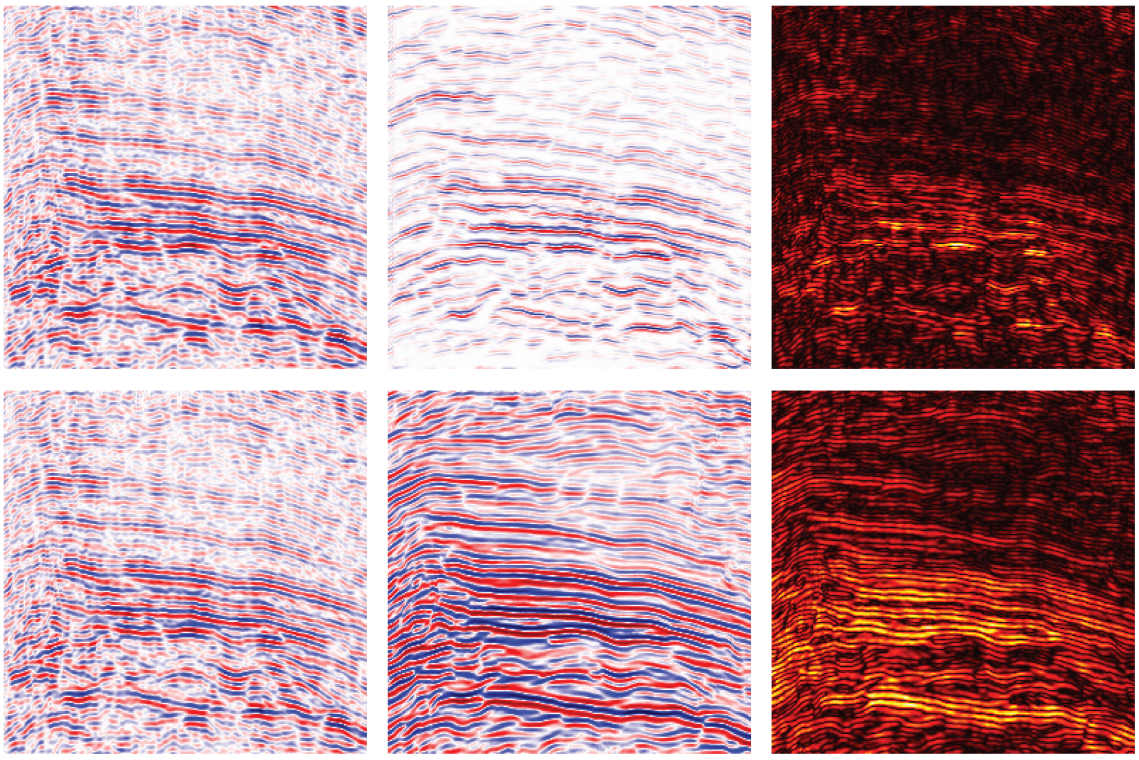


Figura 5.25: Espectro de amplitude em dados reais – Exemplo 1: as curvas em vermelho e em verde mostram os espectros de Fourier dos dados reais e dos dados gerados. Na primeira linha, apresentamos os espectros obtidos com o Modelo 5 e na segunda linha com o Modelo 22.



(a) (b) (c)

Figura 5.26: Resultados da predição em dados reais – Exemplo 2: (a) é o dado real, (b) é o dado gerado e (c) é a diferença entre o real e o dado gerado. Na primeira linha, apresentamos os resultados utilizando o Modelo 5 e na segunda linha o Modelo 22.

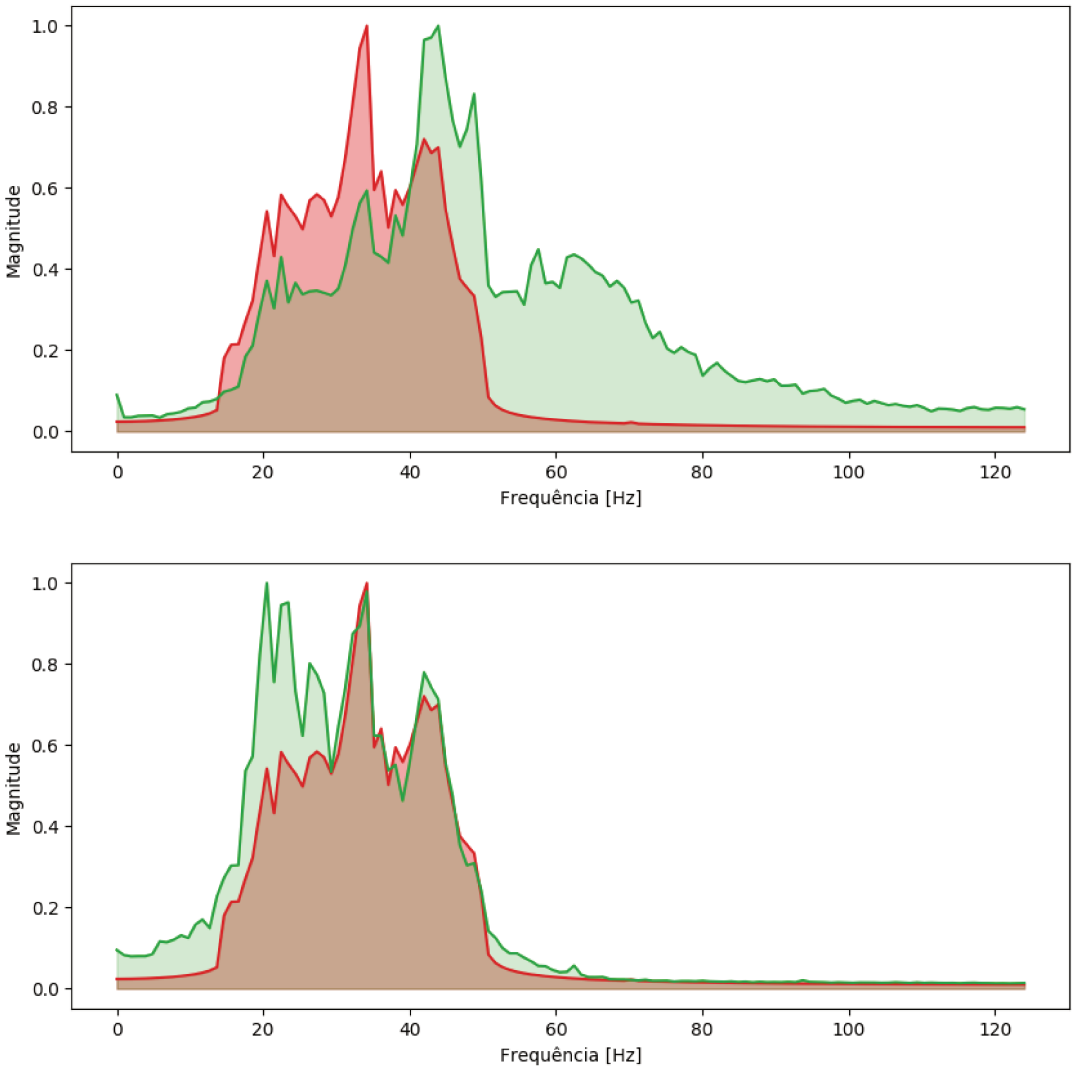
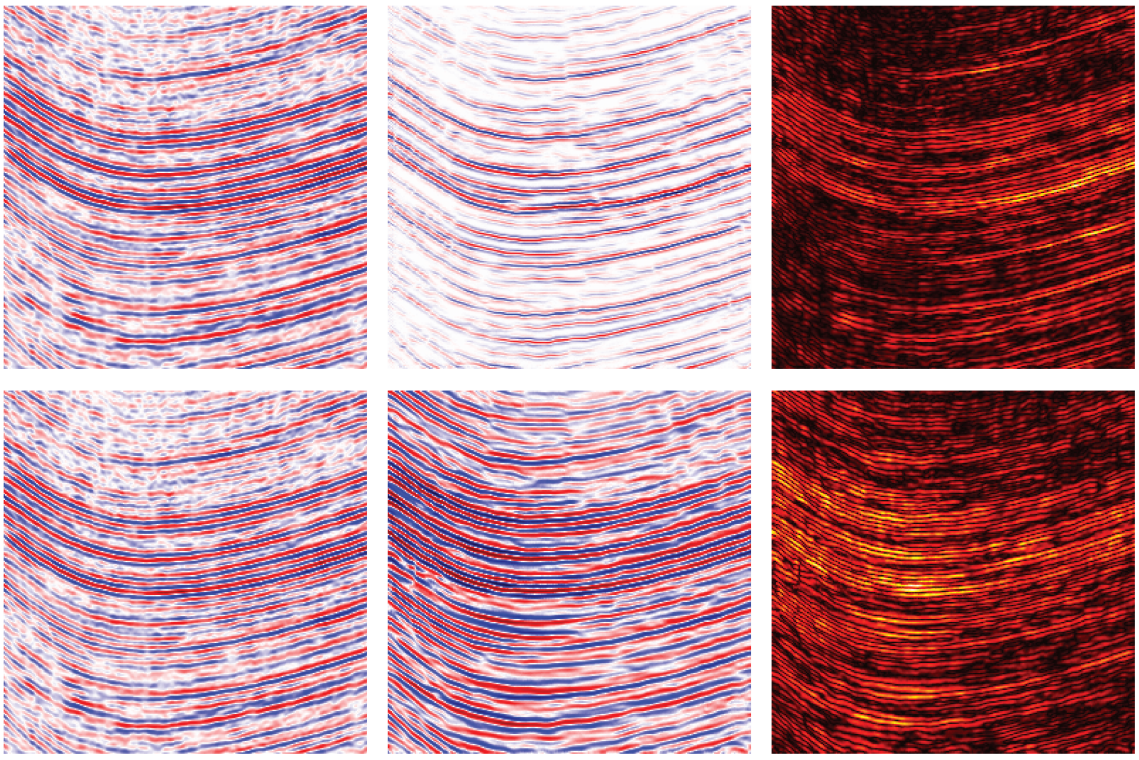


Figura 5.27: Espectro de amplitude em dados reais – Exemplo 2: as curvas em vermelho e em verde mostram os espectros de Fourier dos dados reais e dos dados gerados. Na primeira linha, apresentamos os espectros obtidos com o Modelo 5 e na segunda linha com o Modelo 22.



(a) (b) (c)

Figura 5.28: Resultados da predição em dados reais – Exemplo 3: (a) é o dado real, (b) é o dado gerado e (c) é a diferença entre o real e o dado gerado. Na primeira linha, apresentamos os resultados utilizando o Modelo 5 e na segunda linha o Modelo 22.

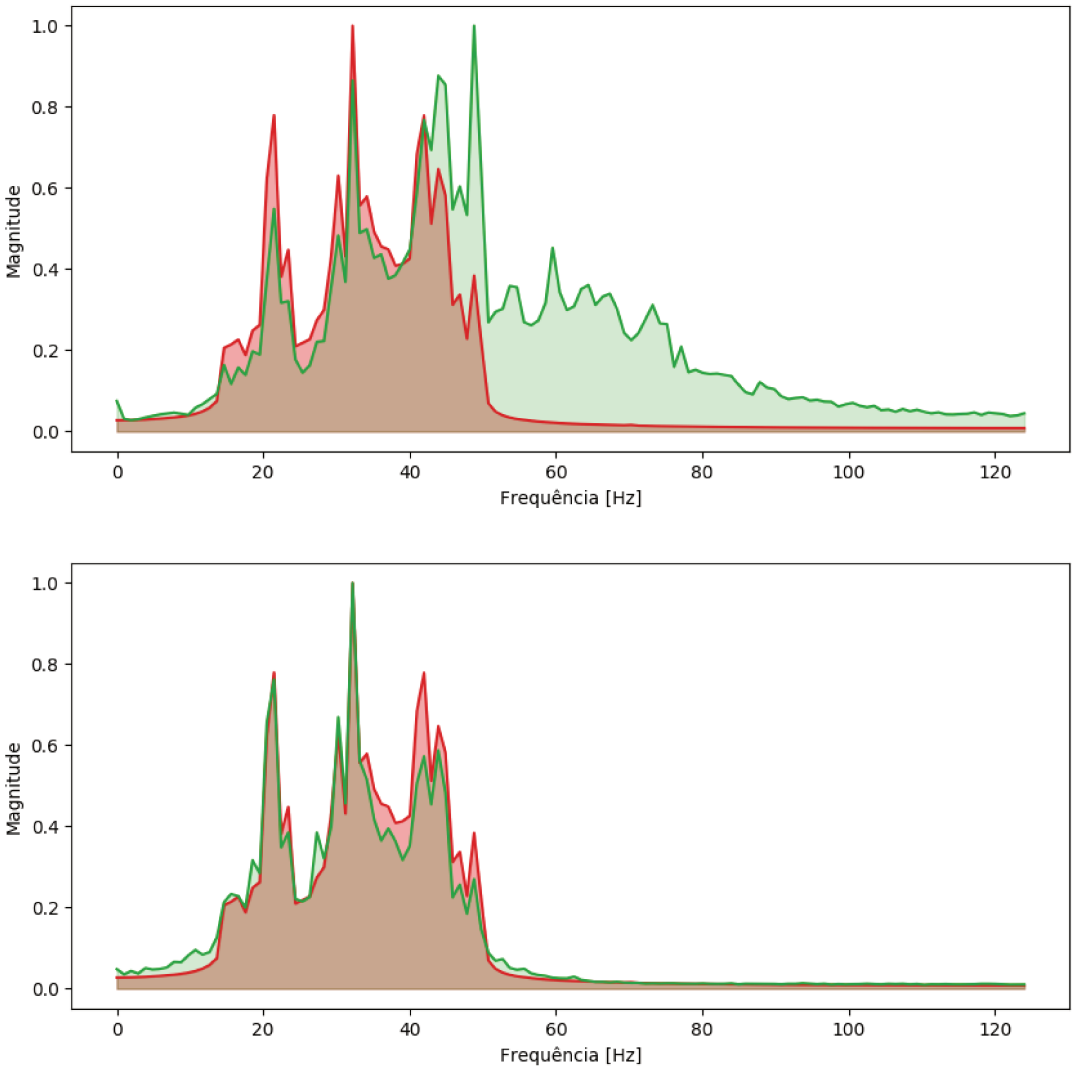


Figura 5.29: Espectro de amplitude em dados reais – Exemplo 3: as curvas em vermelho e em verde mostram os espectros de Fourier dos dados reais e dos dados gerados. Na primeira linha, apresentamos os espectros obtidos com o Modelo 5 e na segunda linha com o Modelo 22.

5.6 Considerações Finais

Após finalizarmos os experimentos, definimos a nossa arquitetura padrão como sendo a Pix2PixHD, tendo em vista o tamanho da rede, o tempo de treinamento e os resultados. Em relação aos modelos, consideramos o Modelo 22 — treinado com o GD2 e função de custo $\mathcal{F}_{\text{MSSSIM}}$ — que obteve acurácia de $77,19\%$. Entretanto, como os experimentos realizados utilizando o GD2 não consideravam o ganho de altas frequências, selecionamos o Modelo 5 como sendo o nosso modelo padrão com ganho de altas frequências, tendo sido treinado com o GD1 que obteve acurácia de $86,65\%$.

Capítulo 6

Conclusões e Trabalhos Futuros

Neste capítulo, recapitulamos nossas descobertas e os principais tópicos discutidos nesta dissertação de Mestrado. Além disso, apresentamos algumas possibilidades para a exploração em trabalhos futuros referentes à geração de dados sintéticos, treinamento de novos modelos e produtização.

Nossas principais contribuições são geradores de dados sísmicos sintéticos — GD1 e GD2 — e abordagens baseadas em GANs para remoção de ruídos e aumento de altas frequências. Com as nossas bases de dados sintéticas, fomos capazes de treinar modelos de GANs, resolvendo assim o problema inicial da escassez de dados. Apesar das bases de dados ter apresentado bons resultados, ressaltamos que os dados não são representativos de todos os tipos de dados sísmicos, podendo ser necessário adaptações de acordo com o alvo da predição e com o problema em questão.

Quanto às arquiteturas de GANs, exploramos diversos modelos baseados em três arquiteturas: Pix2Pix, Pix2PixHD e SPADE. Utilizamos estes modelos para realizar a predição de dados sísmicos sintéticos (dados produzidos pelos geradores propostos e dados da literatura) e dados sísmicos reais. Destacamos que as nossas abordagens são capazes de realizar o treinamento com dados sísmicos brutos, ou seja, não é necessário fazer a conversão dos dados sísmicos para imagens em tons de cinza (e nem tão pouco o contrário). Ressaltamos que, até onde sabemos, nossa abordagem proposta é a primeira a realizar tal treinamento na literatura.

Os nossos resultados, tanto em dados sísmicos sintéticos quanto em dados sísmicos reais, sugerem que as abordagens propostas são promissoras, apresentando melhorias significativas na relação sinal-ruído e no ganho de altas frequências. Os nossos resultados também demonstram que é possível utilizar as GANs como ferramentas de apoio para os especialistas de sísmica, ajudando assim na tomada de decisões e na realização de análises de forma mais rápida e assertiva. Ressaltamos que esse trabalho não visa a substituição de especialistas em sísmica, muito pelo contrário, o especialista representa uma parte fundamental do processo, pois é a pessoa especialista quem irá determinar se os resultados gerados por nosso modelo são úteis. Desse modo, nossos modelos têm como objetivo ajudá-las nas suas tarefas e na redução de processos manuais e repetitivos.

6.1 Questões de Pesquisa

Recapitulamos as principais descobertas organizadas na forma de resposta das questões de pesquisa.

QP1: Para realizar a remoção de ruídos e o ganho de alta frequência, GANs projetadas para imagens são eficazes em dados sísmicos?

Como descrito nas Seções 5.2, 5.3, 5.4, e 5.5, obtivemos resultados superiores aos métodos tradicionais. Entretanto, ocorre a perda de precisão quando realizada a conversão dos dados sísmicos para imagem. Resolvemos esse problema com a modificação das entradas e saídas das redes de imagens para dados sísmicos. Logo, GANs projetadas para imagens podem ser eficientes em dados sísmicos.

QP2: É possível utilizar GANs para a remoção de ruídos de dados sísmicos e o ganho de alta frequência?

Na Seção 5.2 apresentamos o dado gerado pelo gerador da GAN, e quando comparado com o dado sem ruído apresenta alto índice de acerto. Também apresentamos alguns espectros de frequências que demostram que houve ganho de altas frequências. Desse modo, é possível utilizar as GANs para realização de ambas as tarefas propostas neste trabalho.

QP3: Uma GAN treinada em dados sísmicos sintéticos (com e sem ruídos) é capaz de remover ruídos em dados sísmicos reais?

Conforme Seção 5.5, os resultados demostram que é possível utilizar os modelos treinados em dados sintéticos para realizar a predição de dados reais. Entretanto, isso deve ser realizado com o apoio de um especialista e em dados do mesmo domínio daqueles utilizados no treinamento.

6.2 Trabalhos Futuros

O presente trabalho pode ser continuado explorando-se os seguintes aspectos: Gerador de Dados Sísmicos Sintéticos, Arquitetura e Treinamento dos Modelos, e Produtização.

Gerador de Dados Sísmicos Sintéticos

- *Automatização do processo de extração de PSF*: Conforme apresentado na Seção 5.1.1, atualmente temos um conjunto de PSFs limitado, dado que a sua extração deve ser realizada manualmente por um especialista. Diante disso, é importante investigar possíveis caminhos para automatizar esse processo de extração, como também aumentar a quantidade de PSF extraídas para melhorar a variabilidade da base de dados e a representatividade dos dados reais. Uma possível solução para este problema é realizar o treinamento de uma rede para detecção de PSFs. Entretanto, isso requer uma base de dados rotulada o que demandaria muitos dados reais e tempo dos especialistas para realizar a anotação. Apenas esse processo já

seria uma nova frente de pesquisa, sendo este um dos principais motivos pelos quais não exploramos este caminho. Outra possibilidade é aprender o estilo do ruído real para gerar dados sintéticos mais realistas, como proposto por Takemoto et al. [66].

- *Aumento da resolução dos dados gerados pelo GD2*: O GD2 produz dados sísmicos com resolução de 256×256. Como os resultados com o GD1 apresentaram uma melhora com a mudança de resolução (de 256×256 para 512×512), acreditamos que o mesmo pode acontecer para nos resultados com a resolução 512×512 para o GD2. O código disponibilizado no GitHub já é capaz de gerar dados nessa nova resolução, sendo necessário a geração de um nova base de dados e a execução de novos experimentos.

- *Ganho de frequência dos dados gerados pelo GD2*: O treinamento da rede para que gere dados com altas frequências necessita que os dados sintéticos sem ruídos sejam gerados com altas frequências, o que não foi implementado no GD2 para os resultados reportados nesta dissertação. Entretanto, o código disponibilizado no GitHub já possui os parâmetros necessários para a alteração da frequência do dado sem ruído, sendo necessário a geração de uma nova base de dados e a execução de novos experimentos. Essa é uma importante sugestão para trabalhos futuros.

Arquitetura e Treinamento dos Modelos

- *Combinação das funções de custo*: Na Seção 5.4, realizamos alguns experimentos para investigar a contribuição das funções de custo para o modelo. Sugerimos experimentar outras combinações dentre as funções de custos implementadas. Por exemplo, cada uma das funções de custo desenvolvida foi testada separadamente (ou seja, funções de custo nativa da Pix2PixHD + uma das nossas), e como resultados tivemos um aumento significativo na acurácia quando utilizamos as funções $\mathcal{F}_{\text{SEISMIC}}$ e $\mathcal{F}_{\text{MSSSIM}}$ individualmente. Esperamos que a combinação apresente um resultado melhor.

- *Ponderação das funções de custo*: Outro ponto que pode ser explorado é a ponderação das funções de custo, pois nos nossos experimentos utilizamos pesos 1 e 10, mas outros valores podem ser investigados ou aprendidos via *grid search*, por exemplo. Além disso, sugerimos realizar um estudo para o entendimento de qual função de custo é mais importante e em qual etapa do treinamento (início, meio e fim). Por exemplo, temos uma hipótese não explorada de que pode ser melhor utilizar a função de custo para ajuste de frequência mais no fim do treinamento, em que a rede já tenha aprendido as características gerais dos dados.

- *Treinamento por progressão da resolução*: realizar o treinamento por etapa, iniciando de uma resolução inferior e aumentando gradativamente a resolução a cada treinamento, com o intuito de que a rede aprenda primeiro a estrutura geral dos dados e depois os detalhes finos.

Produtização

- *Predição de dados sísmicos inteiros*: Os nossos modelos realizam a predição na mesma resolução na qual foi treinada, logo ainda não é possível predizer todo o dado sísmico em uma única vez, sendo necessário fazer separação dos dados em *patches*, realizar a predição destes e, por fim, juntar os *patches* (ou realizar a interpretação de cada um deles separadamente). Assim, para permitir o uso dos nossos modelos por especialistas em Sísmica é mais apropriado a análise do dado sísmico por inteiro. Uma possível solução para esse problema seria a remoção das camadas totalmente conectadas, deixando a rede totalmente convolucional, conforme realizado por Coelho et al. [10].

- *Desenvolvimento de software ponta a ponta*: Seguindo a mesma linha do item anterior, também é sugerida a construção de um software que consiga realizar todo o processo de forma transparente para o especialista.

6.3 Publicações e Distinções

- Jonlenes Castro, Marcelo Silva, Tiago A. Coimbra, and Sandra Avila, "Noise Removal with Generative Adversarial Networks", in: *Joint SBGf-SEG Workshop on Machine Learning* (em revisão). Nesta submissão, discutimos os resultados iniciais das abordagens baseadas em GANs, Pix2Pix e Pix2PixHD.

- No momento, estamos preparando a submissão dos resultados reportados para o periódico *Geophysics*, com fator de impacto 2.793.

- Bolsista selecionado do QuintoAndar em 2019. A escolha do bolsista foi baseada em desempenho acadêmico e impacto potencial do projeto, em qualquer segmento da Ciência da Computação, sem restrições.

Referências Bibliográficas

[1] Martín Abadi, Ashish Agarwal, e Paul Barham. TensorFlow: Deep convolutional generative adversarial network. `https://www.tensorflow.org/tutorials/generative/dcgan`, 2015. 20

[2] Stephen Alwon et al. Generative adversarial networks in seismic data processing. In *2018 SEG International Exposition and Annual Meeting*. Society of Exploration Geophysicists, 2018. 34, 38

[3] Duhyeon Bang e Hyunjung Shim. Mggan: Solving mode collapse using manifold guided training. *arXiv preprint arXiv:1804.04391*, 2018. 22

[4] Michael Bernico. *Deep learning quick reference: useful hacks for training and optimizing deep neural networks with TensorFlow and Keras*. Packt Publishing Ltd, 2018. 15, 18, 19, 21, 22

[5] Ricardo Biloti. Processamento sísmico. `www.ime.unicamp.br/~biloti/geo/notas.pdf`, 2019. 29

[6] Alceu Bissoto, Eduardo Valle, e Sandra Avila. The six fronts of the generative adversarial networks. *arXiv preprint arXiv:1910.13076*, 2019. 22

[7] Dekuan Chang, Wuyang Yang, Xueshan Yong, e Haishan Li. Generative adversarial networks for seismic data interpolation. In *SEG 2018 Workshop: SEG Maximizing Asset Value Through Artificial Intelligence and Machine Learning, Beijing, China, 17-19 September 2018*, pages 40–43. Society of Exploration Geophysicists and the Chinese Geophysical Society, 2018. 15, 34, 38

[8] Jinsong Chen, Michael Commer, e Gary Michael Hoversten. Integration of seismic and flow-related data for high-resolution reservoir imaging using bayesian models and conditional generative adversarial networks. In *American Geophysical Union*, 2018. 36, 37, 38

[9] Ming-Jun Chen e Alan C Bovik. Fast structural similarity index algorithm. *Journal of Real-Time Image Processing*, 6(4):281–287, 2011. 45

[10] Thamiris Coelho, Lucas Araújo, Tiago Coimbra, Martin Tygel, Sandra Avila, e Edson Borin. Automatic detection of diffraction-apex using fully convolutional networks. In *Int. Congress of the Brazilian Geophysical Society*, 2019. 79

[11] Marcílio Castro de Matos. *Reconhecimento de Padrões Sísmicos Utilizando Análises Tempo-Freqüência*. PhD thesis, PUC-Rio, 2004. 15

[12] Henry den Bok. Sigsbee2a 2d synthetic dataset. `http://www.delphi.tudelft.nl/SMAART/sigsbee2a.htm`, 2001. 67

[13] Haibin Di. Developing a seismic pattern interpretation network (spinet) for automated seismic interpretation. `https://arxiv.org/pdf/1810.08517.pdf`, 2018. 34

[14] Ted E Dohmen. Improved synthetic seismic trace generation using check shot survey data. In *Offshore Technology Conference*, 1982. 32

[15] Jesper S Dramsch, Mikael Lüthje, et al. Deep-learning seismic facies on state-of-the-art cnn architectures. In *2018 SEG International Exposition and Annual Meeting*. Society of Exploration Geophysicists, 2018. 15

[16] Praneet Dutta, Bruce Power, Adam Halpert, Carlos Ezequiel, Aravind Subramanian, Chanchal Chatterjee, Sindhu Hari, Kenton Prindle, Vishal Vaddina, Andrew Leach, et al. 3d conditional generative adversarial networks to enable large-scale seismic image enhancement. *arXiv preprint arXiv:1911.06932*, 2019. 36, 37, 38

[17] Pedro F Felzenszwalb e Daniel P Huttenlocher. Distance transforms of sampled functions. *Theory of Computing*, 8(1):415–428, 2012. 23

[18] Rodrigo Ferreira, Dario Oliveira, e Emilio Brazil. Applying conditional generative adversarial networks for seismic data reconstruction. In *2019 AAPG Annual Convention and Exhibition*, 2019. 15, 34, 38

[19] Rodrigo S Ferreira, Julia Noce, Dario AB Oliveira, e Emilio Vital Brazil. Generating sketch-based synthetic seismic images with generative adversarial networks. *IEEE Geoscience and Remote Sensing Letters*, 2019. 32, 33, 36, 38

[20] Shuwei Gan, Shoudong Wang, Yangkang Chen, Yizhuo Zhang, e Zhaoyu Jin. Dealiased seismic data interpolation using seislet transform with low-frequency constraint. *IEEE Geoscience and remote sensing letters*, 12(10):2150–2154, 2015. 34

[21] Graham Ganssle. Denoising seismic records with image translation networks. `https://csegrecorder.com/assets/pdfs/2018/2018-01-RECORDER-Denoising_Seismic_Records.pdf`, 2018. 15, 16, 36, 37, 38

[22] Zhicheng Geng, Xinming Wu, Yunzhi Shi, e Sergey Fomel. Relative geologic time estimation using a deep convolutional neural network. In *SEG Technical Program Expanded Abstracts 2019*, pages 2238–2242. Society of Exploration Geophysicists, 2019. 42, 46, 47, 66

[23] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, e Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 15, 18, 19, 23, 24

[24] Adam D Halpert et al. Deep learning-enabled seismic image enhancement. In *2018 SEG International Exposition and Annual Meeting*. Society of Exploration Geophysicists, 2018. 36, 37, 38

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, e Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 27, 59

[26] J.W.J. Hosken. Ricker wavelets in their various guises. *First Break*, 6, 1988. doi: 10.3997/1365-2397.1988002. 41

[27] Jonathan Hui. Gan: ways to improve gan performance. `https://towardsdatascience.com/gan-ways-to-improve-gan-performance-acf37f9f59b`, 2018. 26

[28] Luc T Ikelle e Lasse Amundsen. *Introduction to petroleum seismology*. Society of Exploration Geophysicists, 2018. 29

[29] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, e Alexei 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. 16, 23, 25, 40, 43, 47

[30] Edison O Jesus e Roberto Costa Jr. A utilização de filtros gaussianos na análise de imagens digitais. *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics*, 3(1), 2015. 54

[31] Justin Johnson, Alexandre Alahi, e Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016. 25, 26, 44

[32] Harpreet Kaur, Nam Pham, e Sergey Fomel. Seismic data interpolation using cyclegan. In *SEG, 2019*, pages 2202–2206. Society of Exploration Geophysicists, 2019. 34

[33] Santan Kumar, Kiran Kumari, e Ajoy Biswal. Frequency enhancement of seismic data-a comparative study. *CSEG Recorder*, 33(4), 2008. 15

[34] Yann LeCun, Yoshua Bengio, e Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015. 15

[35] Chuan Li e Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European Conference on Computer Vision*, pages 702–716. Springer, 2016. 24

[36] Jerry Li, Aleksander Madry, John Peebles, e Ludwig Schmidt. Towards understanding the dynamics of generative adversarial networks. *arXiv preprint arXiv:1706.09884*, 2017. 15, 21, 22

[37] S. Li, B. Liu, Y. Ren, Y. Chen, S. Yang, Y. Wang, e P. Jiang. Deep-learning inversion of seismic data. *IEEE Transactions on Geoscience and Remote Sensing*, 58(3):2135–2149, 2020. 48

[38] Jesse Lomask, Antoine Guitton, Sergey Fomel, Jon Claerbout, e Alejandro A Valenciano. Flattening without picking. *Geophysics*, 71(4):P13–P20, 2006. 15

[39] Ping Lu, Matt Morris, Seth Brazell, Cody Comiskey, e Yuan Xiao. Using generative adversarial networks to improve deep-learning fault interpretation networks. *The Leading Edge*, 37(8):578–583, 2018. 8, 15, 34, 35, 37, 38

[40] Ping Lu, Yanyan Zhang, Jianxiong Chen, Yuan Xiao, e George Zhao. Enhanced seismic imaging with predictive neural networks for geophysics. *arXiv preprint arXiv:1908.03973*, 2019. 36

[41] Gary Stuart Martin. *The Marmousi2 model: Elastic synthetic data, and an analysis of imaging and AVO in a structurally complex environment.* PhD thesis, University of Houston, 2004. 32

[42] Eric Maskin. Nash equilibrium and welfare optimality. *The Review of Economic Studies*, 66(1):23–38, 1999. 19

[43] Luke Metz, Ben Poole, David Pfau, e Jascha Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016. 22

[44] Mehdi Mirza e Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 23, 34, 36

[45] Lukas Mosser, Wouter Kimman, Jesper Dramsch, Steve Purves, A De la Fuente Briceño, e Graham Ganssle. Rapid seismic domain transfer: Seismic velocity inversion and modeling using deep generative neural networks. In *80th EAGE Conference and Exhibition 2018*, pages 1–5. European Association of Geoscientists & Engineers, 2018. 32, 36, 38

[46] Hisashi Nakahara e Matthew M Haney. Point spread functions for earthquake source imaging: an interpretation based on seismic interferometry. *Geophysical Journal International*, 202(1):54–61, 2015. 41

[47] Dario AB Oliveira, Rodrigo S Ferreira, Reinaldo Silva, e Emilio Vital Brazil. Improving seismic data resolution with deep generative networks. *IEEE Geoscience and Remote Sensing Letters*, 16(12):1929–1933, 2019. 15, 34, 37, 38

[48] Emin Orhan e Xaq Pitkow. Skip connections eliminate singularities. In *ICLR*, 2018. 24

[49] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, e Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019. 15, 18, 27, 28, 29, 40, 44

[50] Francesco Picetti, Vincenzo Lipari, Paolo Bestagini, e Stefano Tubaro. A generative adversarial network for seismic imaging applications. In *88th Society of Exploration Geophysicists International Exposition and Annual Meeting, SEG 2018*, pages 2231–2235, 2018. 32, 36, 38

[51] Francesco Picetti, Vincenzo Lipari, Paolo Bestagini, e Stefano Tubaro. Seismic image processing through the generative adversarial network. *Interpretation*, 7(3):SF15–SF26, 2019. 32, 37, 38

[52] Milton J Porsani. Seismic trace interpolation using half-step prediction filters. *Geophysics*, 64(5):1461–1467, 1999. 34

[53] Alec Radford, Luke Metz, e Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016. 19, 21

[54] Geologic Resources. Seismic reflection description. `http://www.geologicresources.com/seismic_reflection_method.html`, 2018. 30

[55] Alan Richardson. Generative adversarial networks for model order reduction in seismic full-waveform inversion. *arXiv preprint arXiv:1806.00828*, 2018. 36, 38

[56] Olaf Ronneberger, Philipp Fischer, e Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 24

[57] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, e Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016. 22

[58] scikit learn. Neural network models (supervised). `https://scikit-learn.org/stable/modules/neural_networks_supervised.html`, 2019. 19

[59] Ali Siahkoohi, Rajiv Kumar, e F Herrmann. Seismic data reconstruction with generative adversarial networks. In *80th EAGE Conference and Exhibition 2018*, pages 2202–2206, 2018. 8, 34, 35, 38, 47

[60] Thalles Silva. A short introduction to generative adversarial networks. *by sthalles. github.io.[Online*, 2017. 20

[61] Karen Simonyan e Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 26

[62] Kyle Spikes e Jack Dvorkin. Pseudo-well and synthetic seismic data generation. In *2004 SEG Annual Meeting*. Society of Exploration Geophysicists, 2004. 32, 33

[63] Simon Spitz. Seismic trace interpolation in the fx domain. *Geophysics*, 56(6):785–794, 1991. 34

[64] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, e Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 24

[65] Tracy J. Stark. Unwrapping instantaneous phase to generate a relative geologic time volume. In *SEG Technical Program Expanded Abstracts 2003*, pages 1707–1710. Society of Exploration Geophysicists, 2003. 15

[66] Naomi Takemoto, Lucas de M Araújo, Tiago A Coimbra, Martin Tygel, Sandra Avila, e Edson Borin. Enriching synthetic data with real noise using neural style transfer. In *Int. Congress of the Brazilian Geophysical Society*, 2019. 78

[67] The SEG Wiki. Seismic data analysis, 2019. `https://wiki.seg.org/wiki/Seismic_Data_Analysis`. 14

[68] Charles F Van Loan e Gene H Golub. *Matrix computations*. Johns Hopkins University Press Baltimore, 1983. 23, 45

[69] Underground Vaults. Seismic data storage. `https://www.undergroundvaults.com/resources/seismic-data/`, 2018. 29

[70] Benfeng Wang, Ning Zhang, Wenkai Lu, e Jialin Wang. Deep-learning-based seismic data interpolation: A preliminary result. *Geophysics*, 84(1):V11–V20, 2018. 34

[71] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, e Bryan 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, 2018. 15, 25, 26, 27, 28, 40, 43, 45, 59

[72] Zhou Wang, Eero P Simoncelli, e Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003. 45, 46

[73] Hao Wu e Bo Zhang. A deep convolutional encoder-decoder neural network in assisting seismic horizon tracking. *arXiv preprint arXiv:1804.06814*, 2018. 15

[74] Xinming Wu, Yunzhi Shi, Sergey Fomel, e Luming Liang. Convolutional neural networks for fault interpretation in seismic images. In *SEG, 2018*, pages 1946–1950. Society of Exploration Geophysicists, 2018. 15, 32, 33, 35, 37, 38, 42, 52

[75] Wei Xiong, Xu Ji, Yue Ma, Yuxiang Wang, Nasher M AlBinHassan, Mustafa N Ali, e Yi Luo. Seismic fault detection with convolutional neural network. *Geophysics*, 83 (5):O97–O103, 2018. 34, 35

[76] Fangshu Yang e Jianwei Ma. Deep-learning inversion: A next-generation seismic velocity model building method. *Geophysics*, 84(4):R583–R599, 2019. 36

[77] Xin Yi, Ekta Walia, e Paul Babyn. Generative adversarial network in medical imaging: A review. *Medical image analysis*, page 101552, 2019. 24

[78] Oz Yilmaz. *Seismic Data Analysis: Processing, Inversion, and Interpretation of Seismic Data*. Society of Exploration Geophysicists, 2001. 14, 29, 30, 41

[79] Yang Yu, Zhiqiang Gong, Ping Zhong, e Jiaxin Shan. Unsupervised representation learning with deep convolutional neural network for remote sensing images. In *International Conference on Image and Graphics*, pages 97–108. Springer, 2017. 36

[80] Hongliu Zeng, Milo M. Backus, Kenneth T. Barrow, e Noel Tyler. Stratal slicing, part i: realistic 3-d seismic model. *Geophysics*, 63(2):502–513, 1998. 15

[81] Guoyin Zhang, Zhizhang Wang, e Yangkang Chen. Deep learning for seismic lithology prediction. *Geophysical Journal International*, 215(2):1368–1387, 2018. 36, 37, 38

[82] Xiaopu Zhang, Shuai Zhang, Jun Lin, Feng Sun, Xi Zhu, Yang Yang, Xunqian Tong, e Hongyuan Yang. An efficient seismic data acquisition based on compressed sensing architecture with generative adversarial networks. *IEEE Access*, 7:105948–105961, 2019. 15, 38

[83] Yanyan Zhang, Ping Lu, Hua Yu, e Stan Morris. Enhancement of seismic imaging: An innovative deep learning approach. *arXiv preprint arXiv:1909.06016*, 2019. 34, 36, 37, 38

[84] Yuxuan Zhang. Image generation with dcgan. `http://www.timzhangyuxuan.com/project_dcgan`, 2016. 21

[85] Jun-Yan Zhu, Taesung Park, Phillip Isola, e Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 34

Apêndice A

Fluxogramas

O fluxograma de dados, conforme apresentado na Figura A.1, inicia-se no desenvolvimento do gerador de dados, processo que foi acompanhado por um especialista em sísmica que analisa se os dados são representativos dos dados reais. Em seguida, geramos a base de dados (treino, validação e teste) utilizando o gerador. Para a base gerada, é analisado se ela atende os requisitos para realização do treinamento do modelo (por exemplo, quantidade de dados, qualidade, variabilidade). Com a base de dados gerada e validada, avançamos para o fluxograma de treinamento.

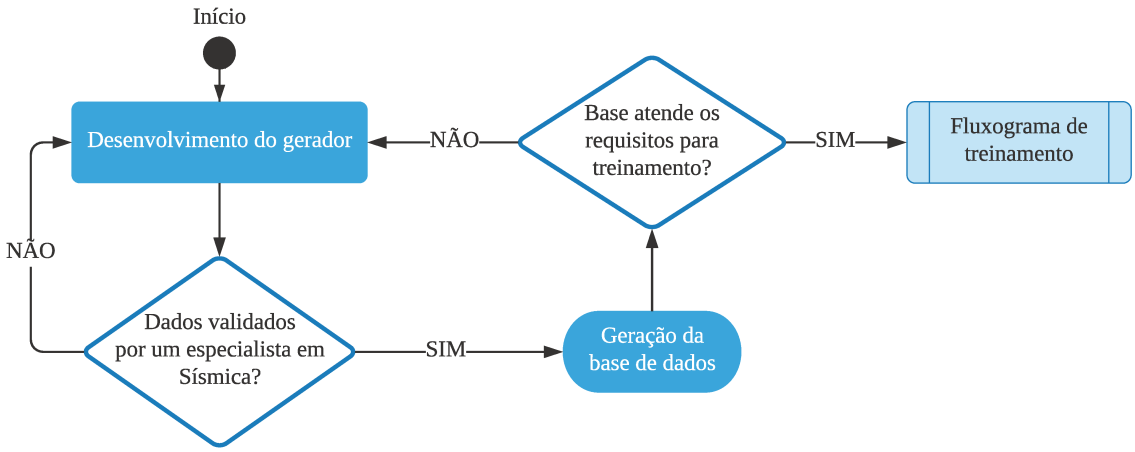


Figura A.1: Fluxograma de dados.

Figura A.2 apresenta o fluxograma de treinamento, onde o primeiro passo consiste na seleção da arquitetura de GAN. Neste trabalho as arquiteturas escolhidas foram a Pix2Pix, Pix2PixHD e SPADE. O passo seguinte consiste na preparação do ambiente para o treinamento desta rede, isto é, a criação e configuração de uma máquina, visto que para cada arquitetura há diferentes requisitos (por exemplo, números de GPUs, memória RAM). O próximo passo consiste na adaptação da arquitetura escolhida, conforme apresentado nas seções 4.2.1, 4.2.2 e 4.2.3. Em seguida, selecionamos os hiperparâmetros, uma etapa de fundamental importância para o treinamento de um bom modelo, sendo este um dos desafio para o treinamento do modelo de GAN. Por fim, realizamos o treinamento, a predição da base de dados de validação e avançamos para o fluxograma de avaliação.

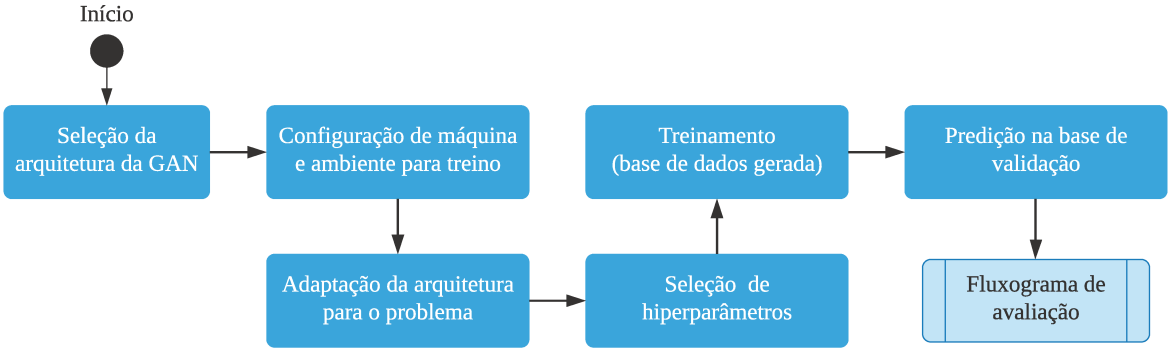


Figura A.2: Fluxograma de treinamento.

O fluxograma de avaliação dos dados preditos, conforme Figura A.3, utiliza os dados gerados na última etapa do fluxograma de treinamento para calcular as medidas de avaliação, como também utiliza outras bases de dados que não foram utilizadas no processo de treinamento. O primeiro passo consiste em calcular essas medidas de avaliação e verificar se todas elas apresentam resultados satisfatórios. Se o modelo for aprovado nesta avaliação, realizamos a predição e validação de dados sintéticos disponíveis na literatura. Estes dados foram escolhidos por serem mais difíceis (considerando sua estrutura geológica) para que o modelo realizasse a sua predição quando comparado aos dados gerados neste trabalho. Feita essa avaliação, avançamos para a predição de dados reais, onde também devem ser calculadas as medidas de avaliação (apenas as cabíveis, visto que os dados reais não possuem o dado sem ruido para realizar a comparação), e a validação por um especialista em sísmica. Após esta validação, temos uma versão do modelo pronto para ser testado.

Por fim, seguimos ao fluxograma de utilização do modelo para remoção de ruídos e ganho de altas frequências de dados sísmicos. Conforme apresentado na Figura A.4, o primeiro caminho inicia com a conversão do dado sísmico para o domínio da imagem. Em seguida, a imagem ruidosa será recebida pelo gerador que produzirá uma imagem potencialmente sem ruídos, que será reconvertida para dado sísmico e, por fim, utilizada para interpretação. Quanto ao segundo modo, o mais simples e mais utilizado nesse trabalho, a rede já recebe o dado sísmico bruto diretamente, não sendo necessário executar os passos de conversão para imagem e vice-versa.

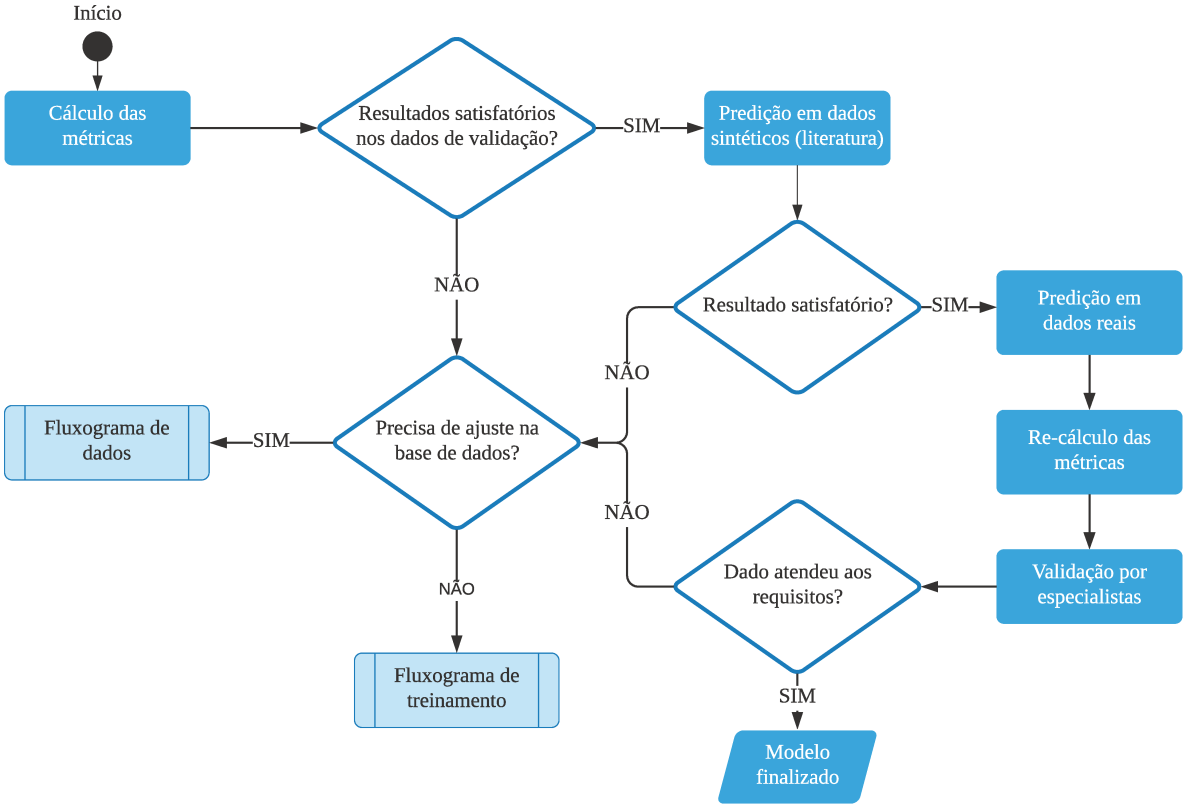


Figura A.3: Fluxograma de avaliação.

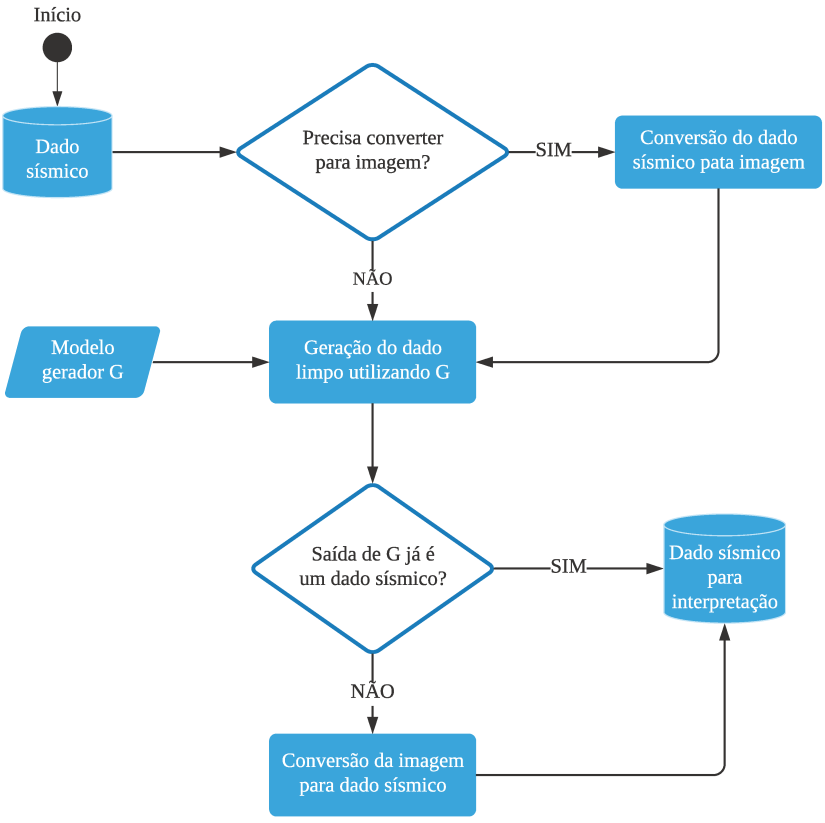


Figura A.4: Fluxograma de utilização do modelo.