Deep convolutional GANs as parameterization method in data assimilation

Presented by:
Paulo Henrique Ranazzi, PhD student
ranazzi@usp.br

Marcio Augusto Sampaio Pinto, Advisor
marciosampaio@usp.br

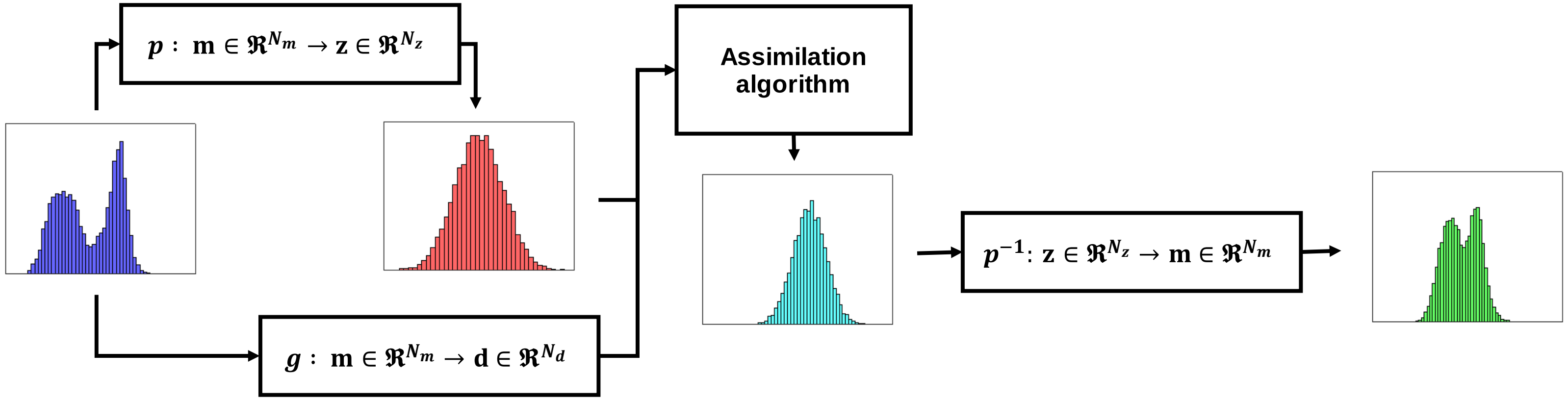
June 01, 2022

Outline

- **Introduction**
- **Generative adversarial networks**
- **Experiment 1: GANs evaluation**
- **Experiment 2: Assimilation**
- **Conclusions**
- **Acknowledgements**
- **References**

Introduction

- Popular ensemble-based methods rely on Gaussian assumption.
- Many parameters have non-Gaussian behavior
  - Categorical facies
  - Multilevel uncertainties
- The assimilation Vanishes its original distribution.
- How to handle with the non-Gaussianity in ensemble-methods?
  - Parameterization techniques

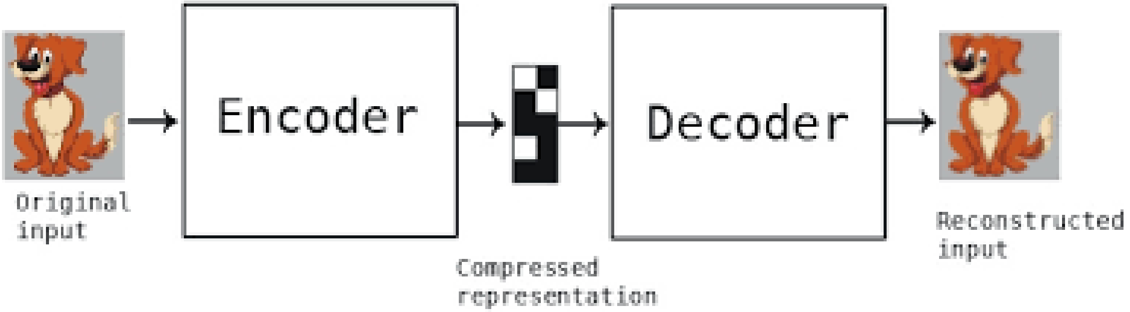


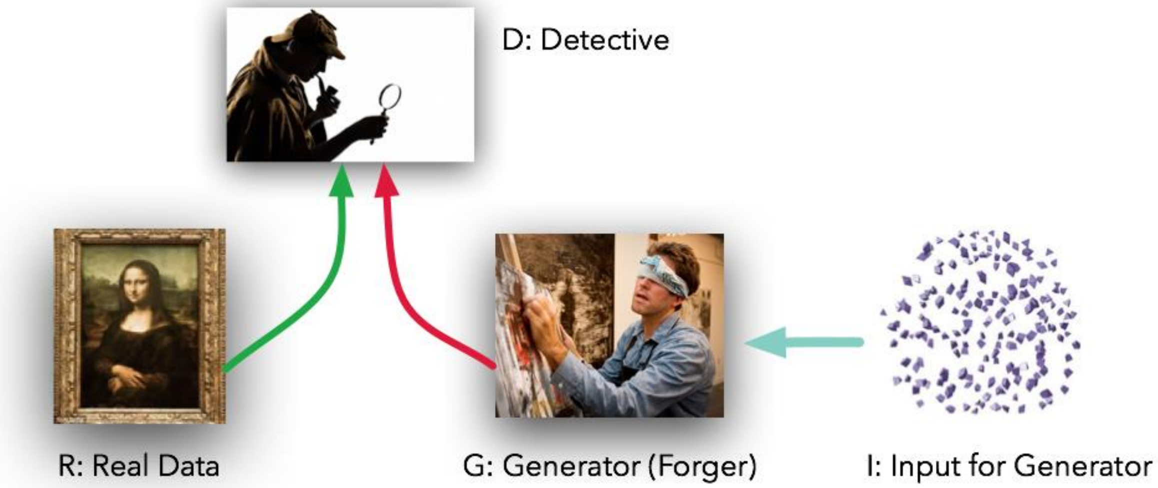
Introduction

- Several parameterization methods.
- Deep learning techniques:
  - Autoencoder (Kim et al., 2020)
  - CNN-PCA (Liu and Durlofsky, 2021)
  - GANs (Canchumuni et al., 2021; Zhang et al., 2022)
- Comparison between GAN and VAE (Bao et al., 2022):
  - VAE performed better in DA
  - GAN resulted in better realizations
- Proceed with investigation of GANs:
  - Evaluate another GAN architecture in DA (Autoencoder Discriminator)

Generative adversarial networks

- GANs (Goodfellow et al., 2014):
  - Composed by a generator and a discriminator:
    - Generator takes a random vector and try to generate new samples.
    - Discriminator try to classify samples as real (from original domain) or fake (generated by generator).
  - Adversarial training:
    - Generator and discriminator must compete against each other.
    - Discriminator is trained to better classify between real and fake.
    - Generator is trained to 'fool' the discriminator.

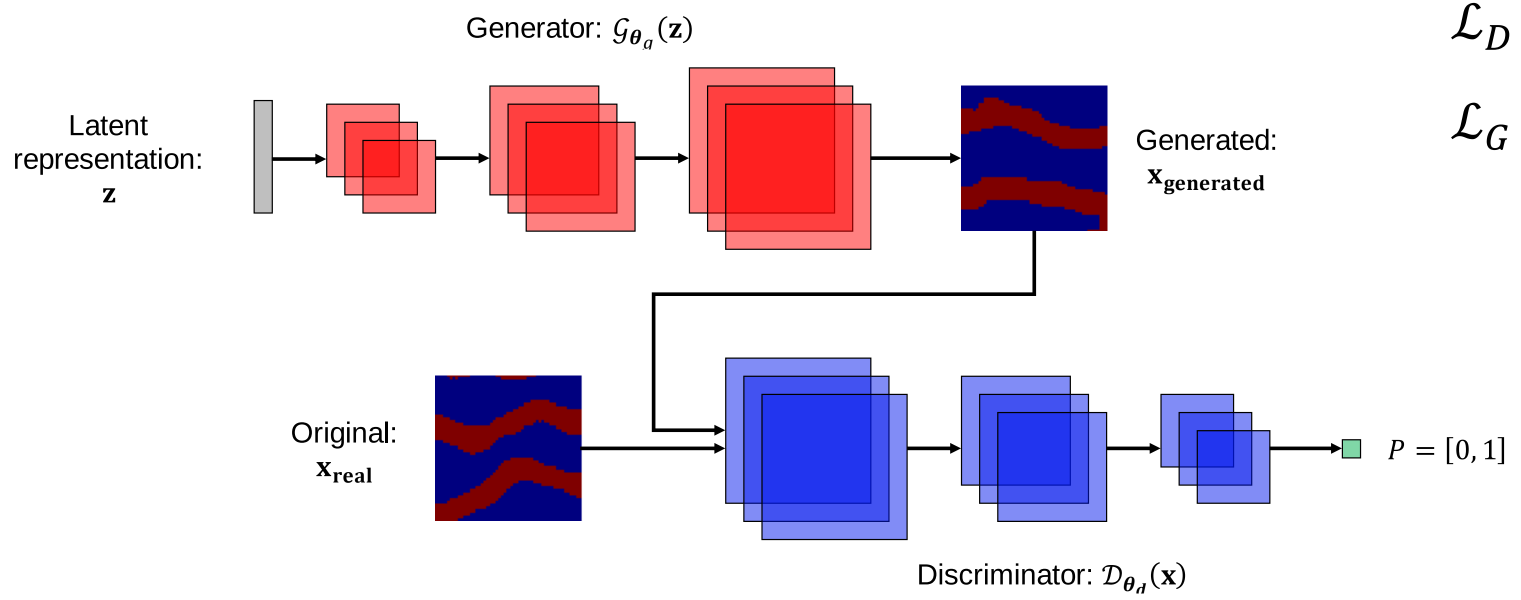




Generative adversarial networks

- DCGAN (Radford et al., 2015):
  - Zero-sum game between $\mathcal{G}$ and $\mathcal{D}$:
  - $\min_{\mathcal{G}} \max_{\mathcal{D}} E_{\mathbf{x}\sim p(\mathbf{x})}[\log \mathcal{D}(\mathbf{x})] + E_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(1-\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)\right)\right]$

In Discriminator training: $\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)$ to 0



$$\mathcal{L}_D = \max_{\mathcal{D}} \log \mathcal{D}(\mathbf{x}) + \log\left(1-\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)\right)$$

$$\mathcal{L}_G = \min_{\mathcal{G}} \log\left(1-\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)\right)$$

In generator training: $\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)$ to 1

Generative adversarial networks

- Wasserstein GAN (WGAN)(Arjovsky et al., 2017):
  - Approximates Earth Mover Distance (EMD) (or Wasserstein-1 distance).

  - Discriminator as critic (don't have probabilities):
    - Instead of classifying, discriminator estimates the Wasserstein distance between real and generated distributions: $\mathcal{D}(\cdot) = [0, \infty]$
    - Critic maximizes the distance between its output on real and fake samples
    - Generator minimizes critic output for fake samples
    - $\min_{\mathcal{G}} \max_{\mathcal{D}} E_{\mathbf{x}\sim p(\mathbf{x})}[\mathcal{D}(\mathbf{x})] - E_{\mathbf{z}\sim p(\mathbf{z})}\left[\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)\right]$
  - Seeks for convergence of generator.

In Discriminator training: $\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)$ to 0

$$\mathcal{L}_D = \max_{\mathcal{D}} \mathcal{D}(\mathbf{x}) - \mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)$$

$$\mathcal{L}_G = \min_{\mathcal{G}} -\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)$$

clip $\mathcal{D}(-c, c)$

In Generator training: $\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)$ to ∞

Generative adversarial networks

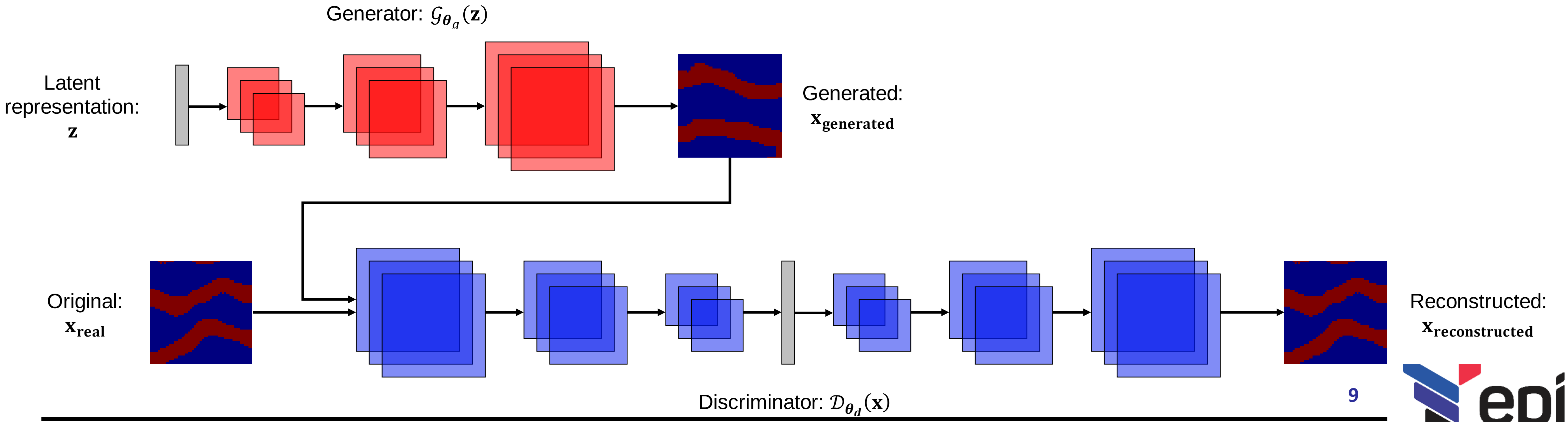
- Wasserstein GAN (WGAN) (Arjovsky et al., 2017):

- Differences:
  - Discriminator output has no constraint (sigmoid in DCGAN because [0, 1])
  - Weight clipping to ensure Lipschitz constraint (w)
  - Need pretrain the discriminator (critic) or train at different rates

- Solved the training failure with cost of convergence speed (Gonog and Zhou, 2019) (also reported in Canchumuni et al., 2021).

Generative adversarial networks

- Boundary Equilibrium GAN (BEGAN) (Berthelot et al., 2017):
  - Autoencoder discriminator (distribution of errors instead distribution of samples)
  - Stable training.
  - Trade-off between image quality and image diversity.
  - Introduced a convergence measure: $\mathcal{M}$



Generative adversarial networks

- Boundary Equilibrium GAN (BEGAN) (Berthelot et al., 2017):
  - Equilibrium when $E_{\mathbf{x}\sim p(\mathbf{x})}[\mathcal{D}(\mathbf{x})] = E_{\mathbf{z}\sim p(\mathbf{z})}\left[\mathcal{D}\left(\mathcal{G}(\mathbf{z})\right)\right]$
  - Discriminator has two objectives:
    - Auto-encode real images
    - Distinguish between real and fake
  - Relax the equilibrium with $\gamma \in [0, 1]$
  - Defining the pixel-wise loss $\mathcal{L}(v) = |v - \mathcal{D}(v)|$
  - Losses:
    - $$\begin{cases} \mathcal{L}_D = \mathcal{L}(\mathbf{x}) - k_t \mathcal{L}\left(\mathcal{G}(\mathbf{z})\right) \\ \mathcal{L}_G = \mathcal{L}\left(\mathcal{G}(\mathbf{z})\right) \\ k_{t+1} = k_t + \lambda_k \left(\gamma \mathcal{L}(\mathbf{x}) - k_t \mathcal{L}\left(\mathcal{G}(\mathbf{z})\right)\right) \end{cases}$$

Diversity ratio

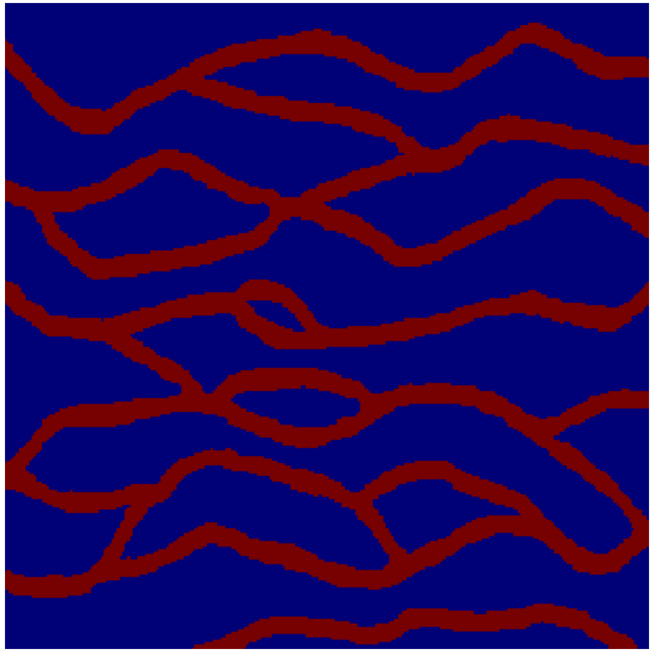
Proportional gain

Samples at varying γ
(Berthelot et al., 2017)

Convergence measure: $\mathcal{M} = \mathcal{L}(\mathbf{x}) + \left|\gamma \mathcal{L}(\mathbf{x}) - \mathcal{L}\left(\mathcal{G}(\mathbf{z})\right)\right|$

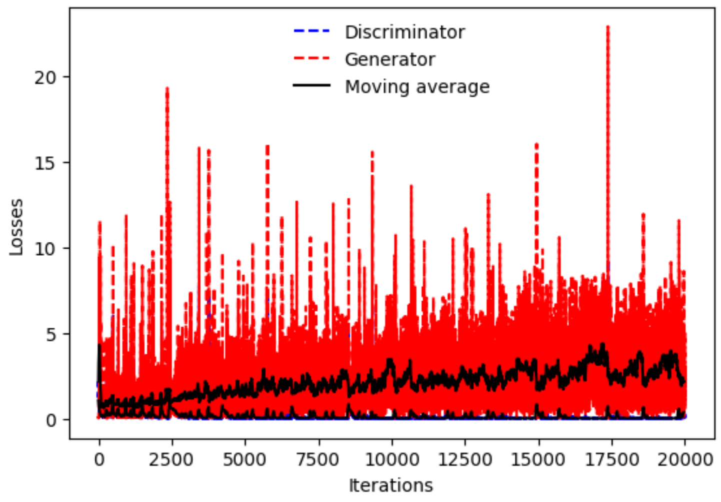
Experiment 1: GANs evaluation

- Channelized reservoir with categorical facies
- 10,000 training samples with SNESIM (Strebelle, 2002)
  - Bao et al. (2022) – 80,000
  - Canchumuni et al. (2021) and Zhang et al. (2022) – 20,000
- 20,000 iterations for all networks (TensorFlow 2.7.0)
- Latent space $N_z = 500$
- 2 channels: [-1, 1] -> final facies is the maximum value Canchumuni et al. (2021)

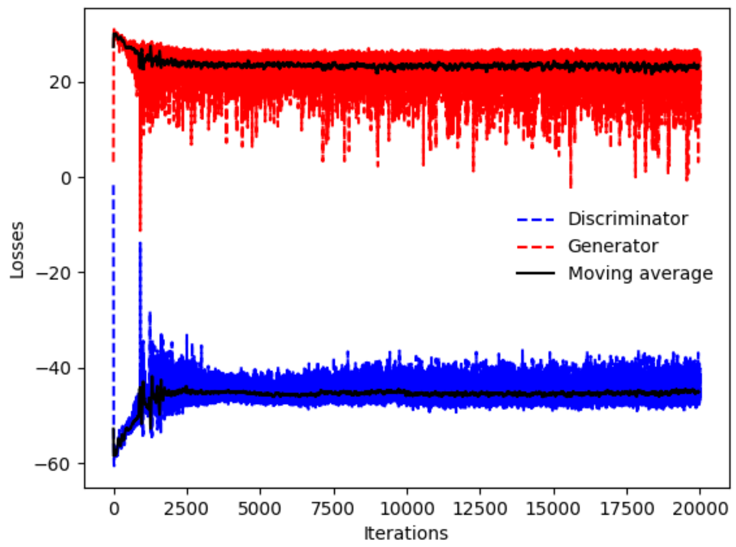


Training image (250x250)

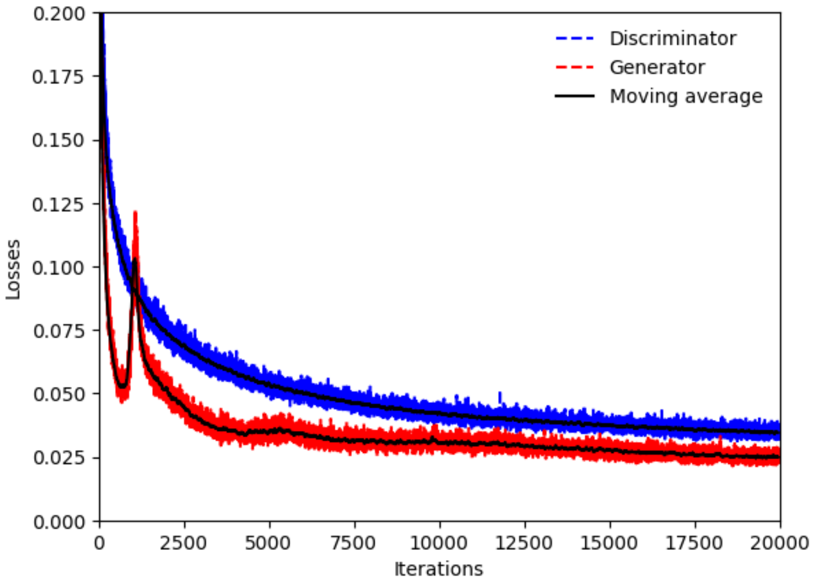
Experiment 1: GANs evaluation



DCGAN



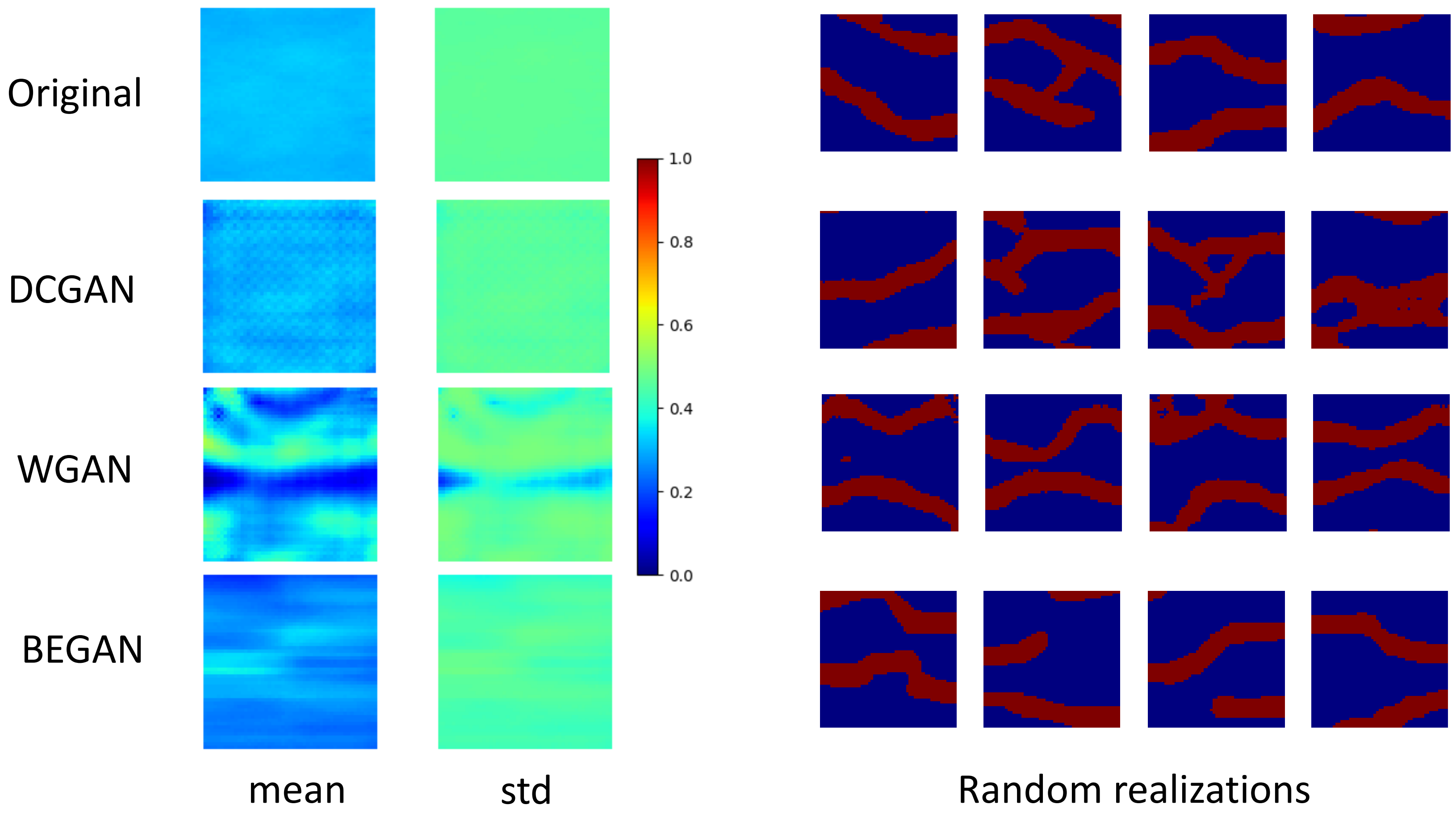
WGAN



BEGAN

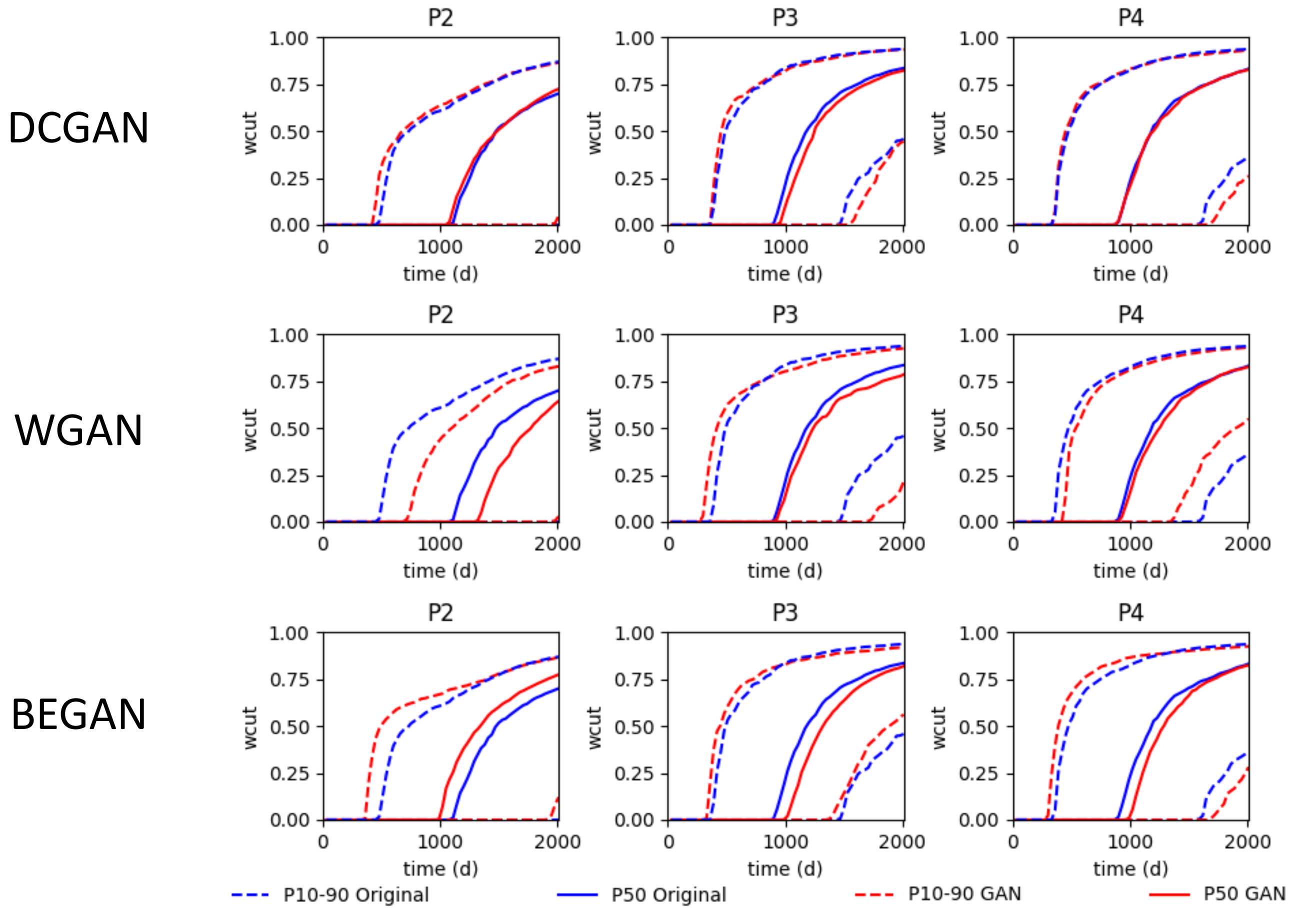
Experiment 1: GANs evaluation

- Random realizations generated with GANs



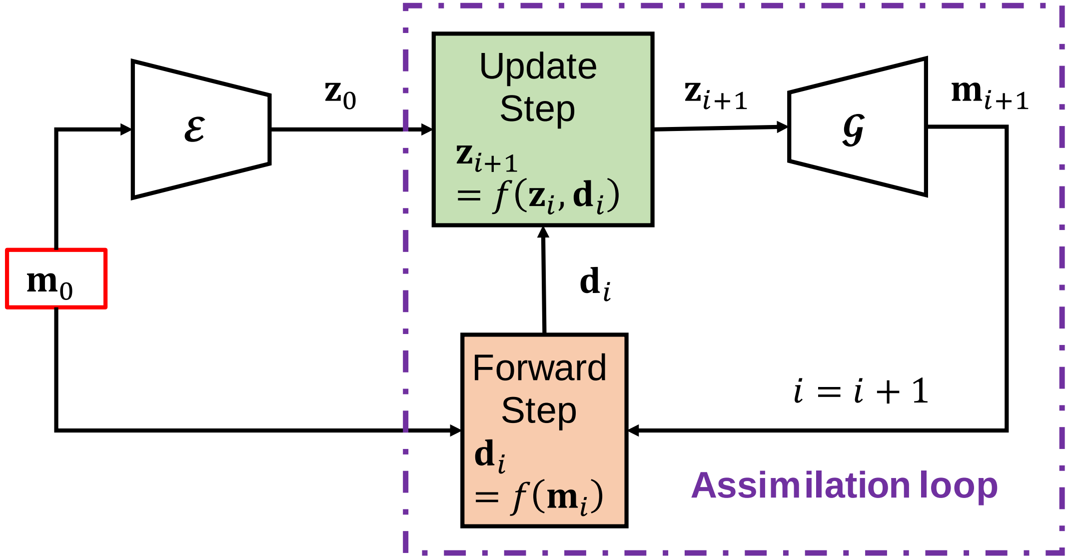
Experiment 1: GANs evaluation

- Percentiles from original and generated ensembles ($N = 200$)

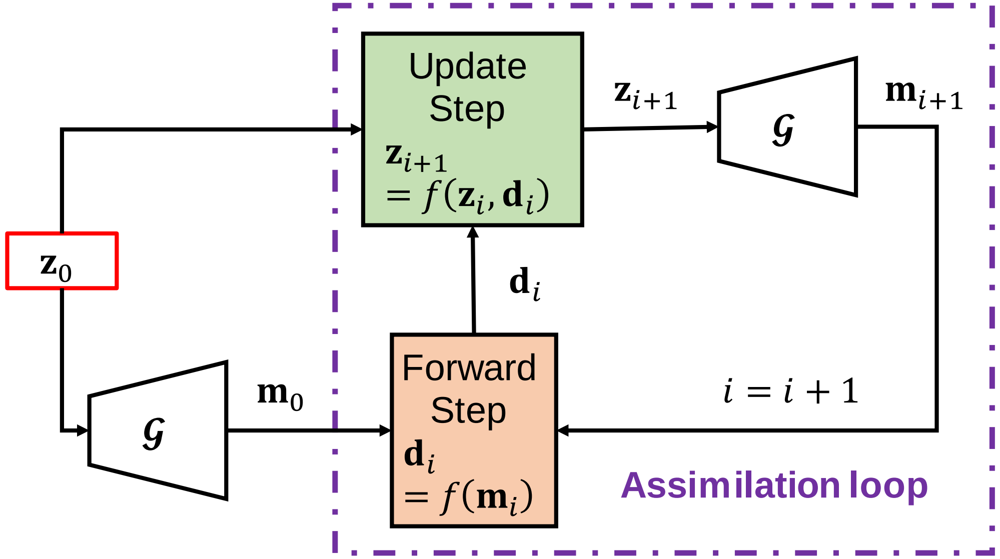


Experiment 2: Assimilation

- Two options:
  - Start by encoding $\{\mathbf{m}_0\}_{j=1}^{N_e}$ to obtain $\{\mathbf{z}_0\}_{j=1}^{N_e}$ (Canchumuni et al., 2021) (encoding)
  - Start by generating the initial ensemble from $\{\mathbf{z}_0\}_{j=1}^{N_e} \sim \boldsymbol{N}(0, \mathbf{I})$ (random)



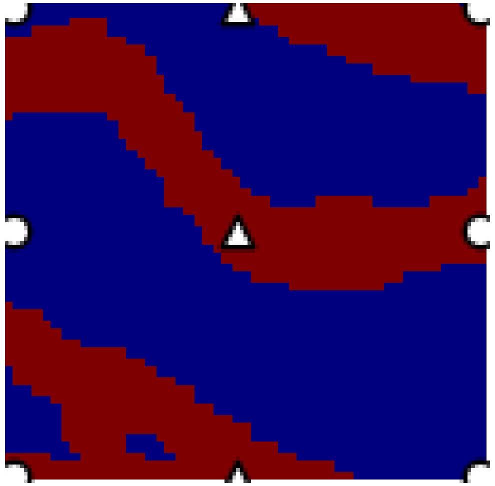
Encoding



Random

Experiment 2: Assimilation

- Test case:



Reference model ($\mathbf{m}_{\text{true}}$)

Size (gridblocks)	51x51x1
Wells	9 (6 producers, 3 injectors)
Data	OPR, WPR, WIR (2500 days) ($N_d = 1005$)
Parameters	Log-Permeability ($N_m = 2601$)
Ensemble size	200
Data assimilation	ES-MDA (Emerick and Reynolds, 2013) with $N_i = \alpha_i = 8$

ES-MDA update step:

$$\mathbf{m}_{j,i+1} = \mathbf{m}_{j,i} + \mathbf{C}_{\mathbf{MD}}(\mathbf{C}_{\mathbf{DD}} + \alpha_i \mathbf{C}_{\mathbf{D}})(\mathbf{d}_{\mathbf{obs},j} - \mathbf{d}_{j,i})$$

$$\sum_{i=1}^{N_i} \alpha_i^{-1} = 1$$

$$\mathbf{d}_{\mathbf{obs},j} \sim (\mathbf{d}_{\mathbf{obs}}, \alpha_i \mathbf{C}_{\mathbf{D}})$$

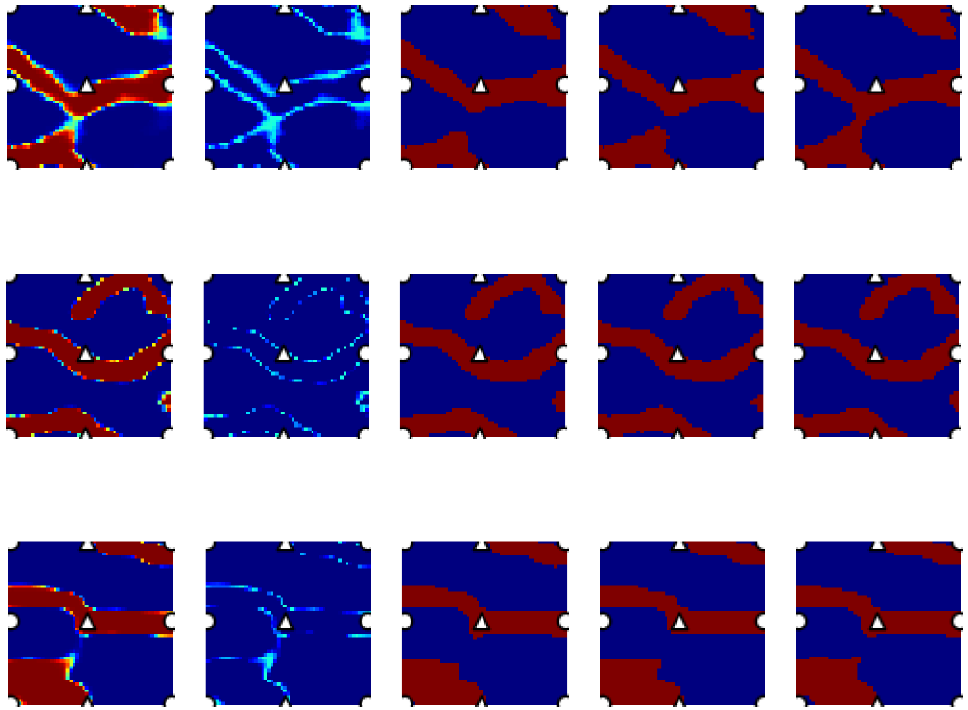
Experiment 2: Assimilation

- Start encoding

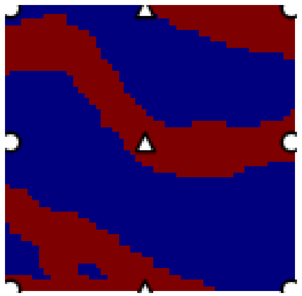
DCGAN

WGAN

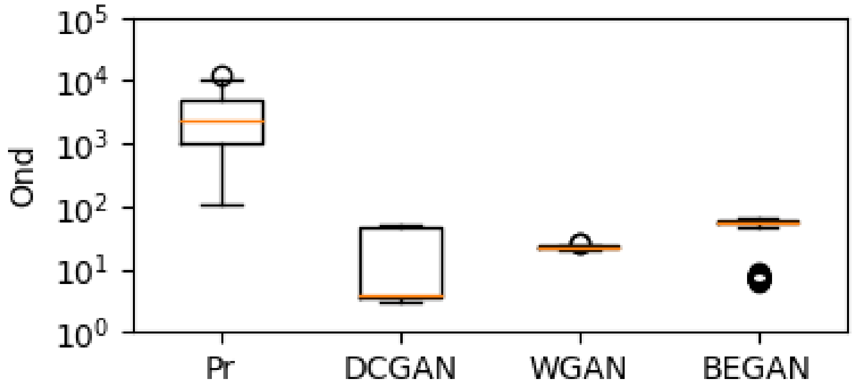
BEGAN



mean std $n = 1$ $n = 2$ $n = 3$



$\mathbf{m}_{\text{true}}$

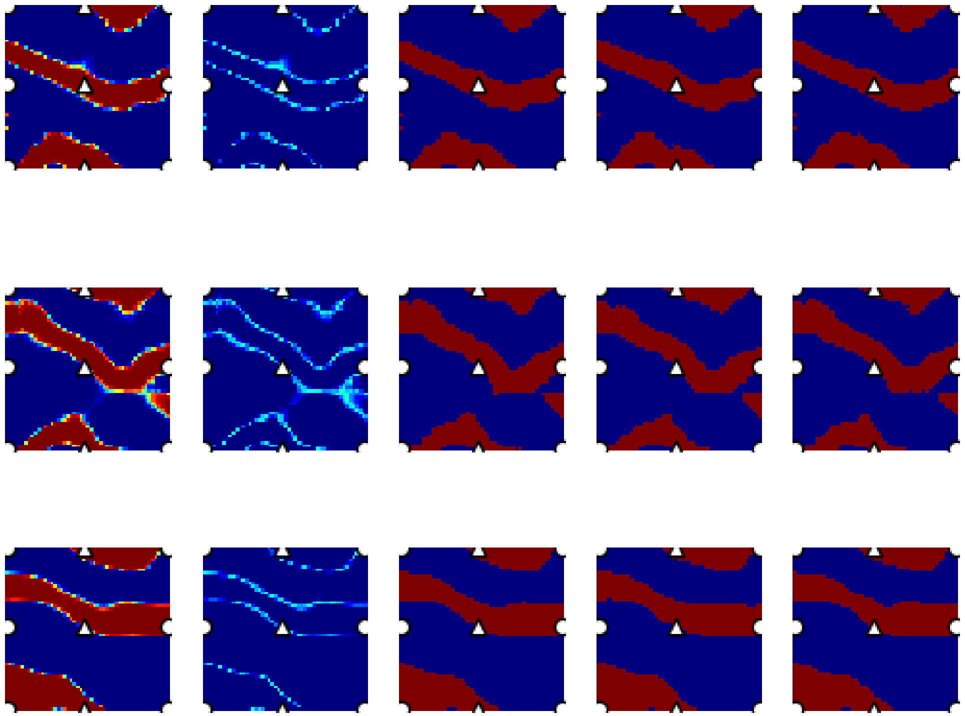


Prior* and posterior data-mismatch for all analyzed cases.

*Prior is the same

Experiment 2: Assimilation

- Start random

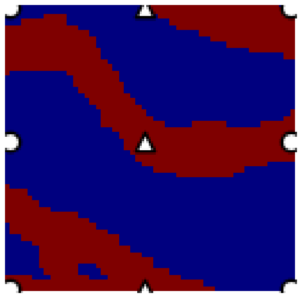


DCGAN

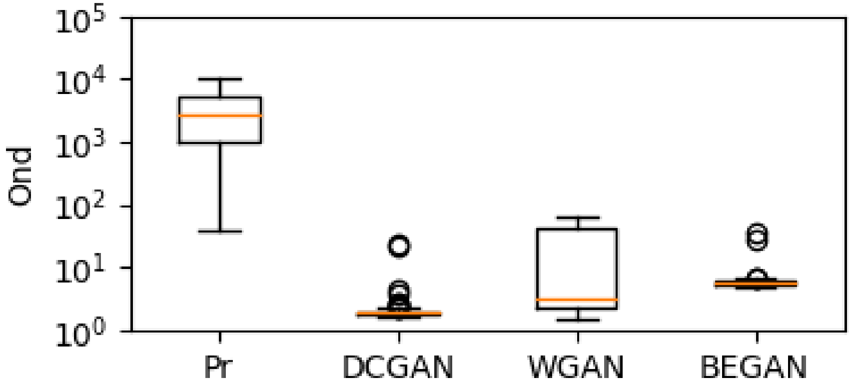
WGAN

BEGAN

mean std $n = 1$ $n = 2$ $n = 3$



$\mathbf{m}_{\mathrm{true}}$



Prior* and posterior data-mismatch for all analyzed cases.

*All cases resulted in similar values

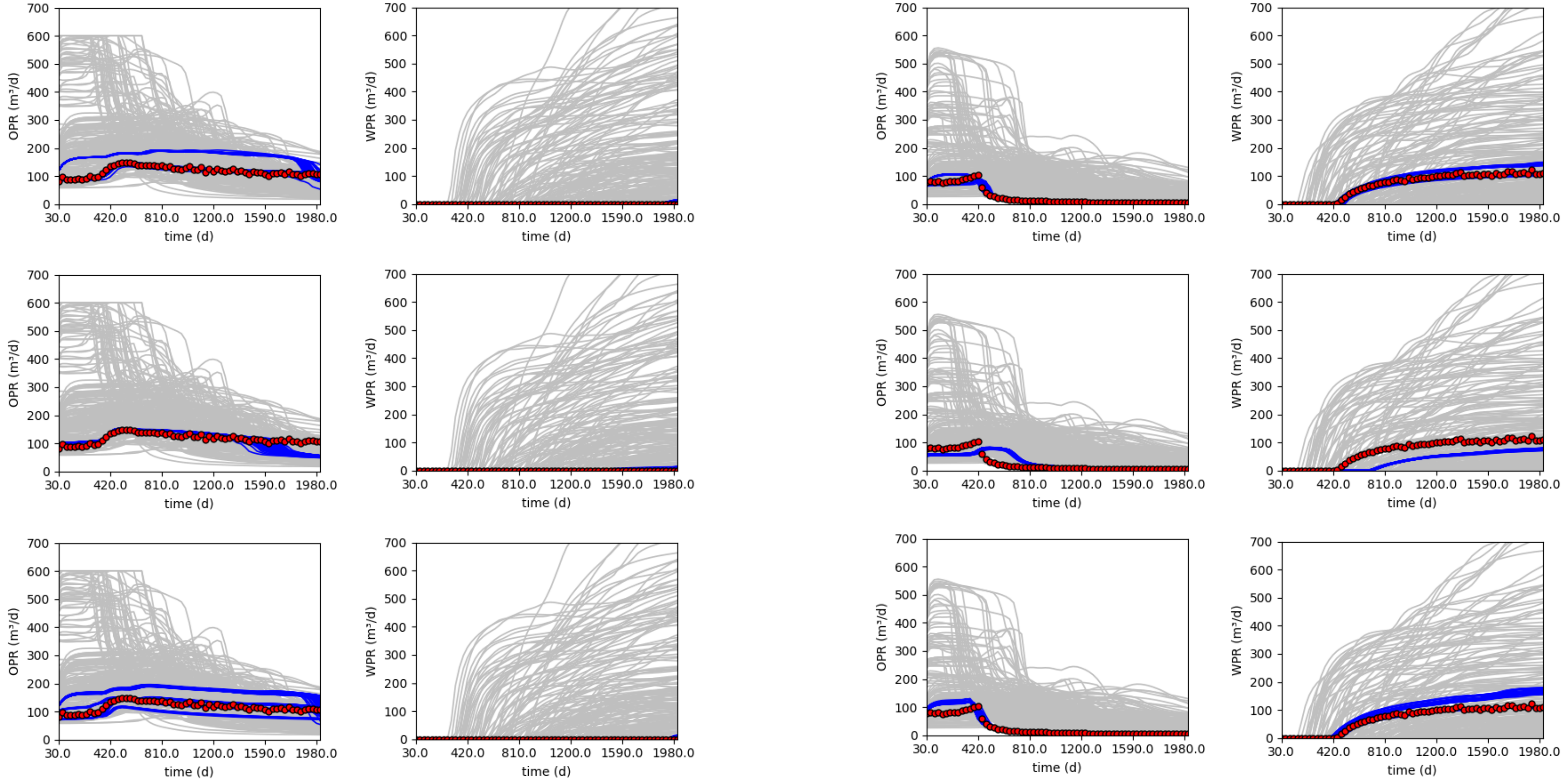
Experiment 2: Assimilation

- Start encoding

DCGAN

WGAN

BEGAN



P2

P4

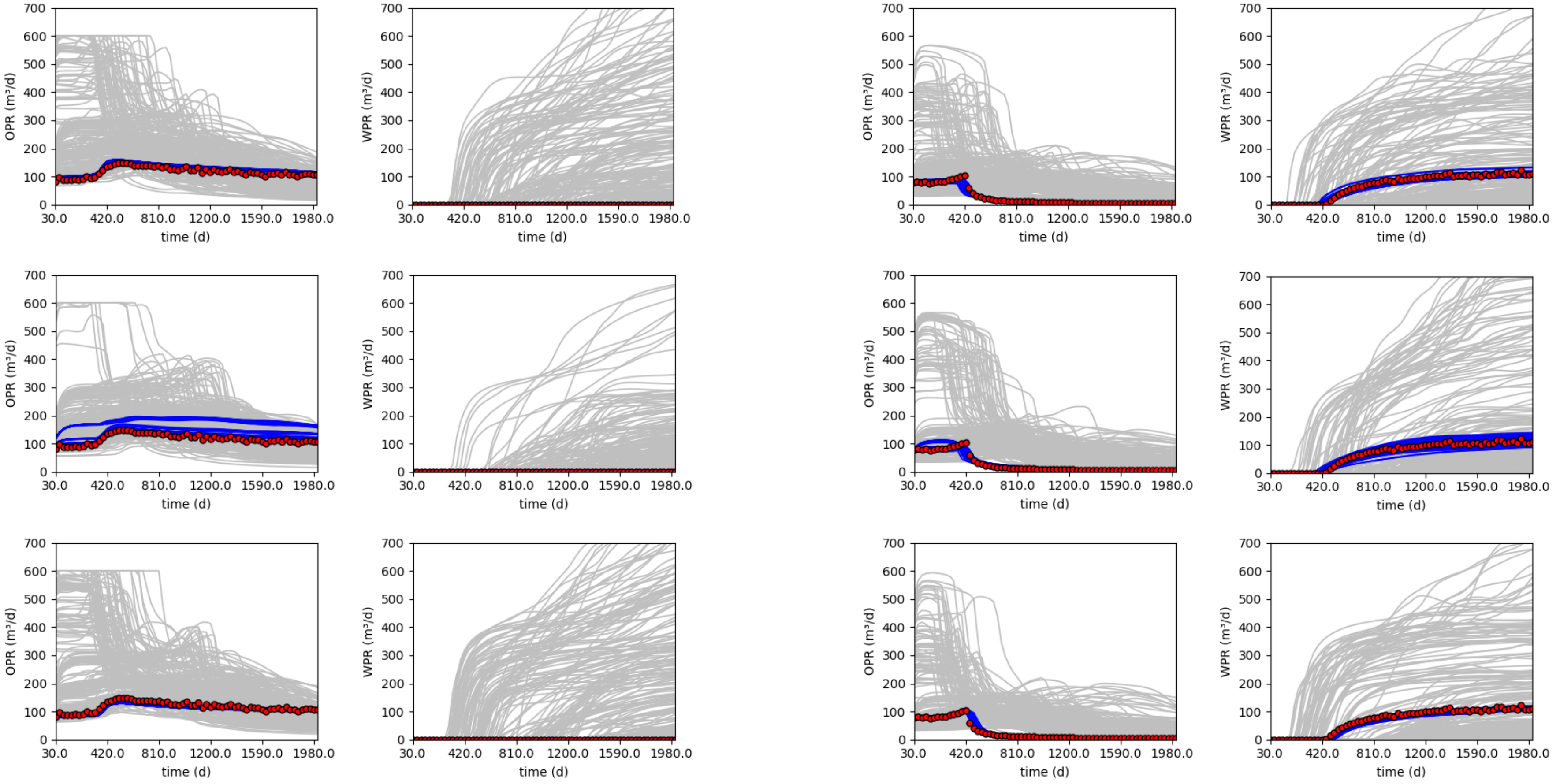
Experiment 2: Assimilation

- Start random

DCGAN

WGAN

BEGAN



P2

P4

Conclusions

- BEGAN resulted in stable training (convergence in loss functions)
- Generated images without discontinuities with ensemble mean and std closer to original ensemble in comparison with WGAN
- Equivalent assimilation results with BEGAN

- Next steps/improvements:
  - Evaluation of hyperparameters effect (γ, N_z)
  - Analysis of different BEGAN architectures (e.g. BEGAN-E) (Xie et al., 2022)
  - Parameterization/generation of 3D reservoirs with GANs stills an open problem

Acknowledgments

References

Kim S., Lee K., Lim J.; Jeong H., Min B., 2020. Development of ensemble smoother-neural network and its application to history matching of channelized reservoirs. Journal of Petroleum Science and Engineering, 191, 107159. https://doi.org/10.1016/j.petrol.2020.107159

Liu Y., Durlofsky L.J., 2021. 3D CNN-PCA: A deep-learning-based parameterization for complex geomodels. Computers & Geosciences, 148, 104676. https://doi.org/10.1016/j.cageo.2020.104676

Bao J., Li L., Davis A., 2022. Variational Autoencoder or Generative Adversarial Networks? A Comparison of Two Deep Learning Methods for Flow and Transport Data Assimilation. Mathematical Geosciences. https://doi.org/10.1007/s11004-022-10003-3

Canchumuni S.W.A., Castro J.D.B., Potratz J., Emerick A.A., Pacheco M.A.C., 2021. Recent developments combining ensemble smoother and deep generative networks for facies history matching. Computational Geosciences, 25, 433-466. https://doi.org/10.1007/s10596-020-10015-0

Zhang K., Yu H.-Q., Ma X.-P., Zhang J.-D., Wang J., Yao C.-J., Yang Y.-F., Sun H., Yao J., Wang J., 2022. Multi-source information fused generative adversarial network model and data assimilation based history matching for reservoir with complex geologies. Petroleum Science, 19, 707-719. https://doi.org/10.1016/j.petsci.2021.10.007

Goodfellow I.J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y., 2014. Generative Adversarial Nets. arXiv:1406.2661 [stat.ML]

Radford A., Metz L., Chintala S., 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv:1511.06434v2 [cs.LG]

References

Arjovsky M., Chintala S., Bottou L., 2017. Wassertein GAN. arXiv:1701.07875 [stat.ML]

Gonog L., Zhou Y., 2019. A Review: Generative Adversarial Networks. Presented in: 2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA). https://doi.org/10.1109/ICIEA.2019.8833686

Berthelot D., Schumm T., Metz L., 2017. BEGAN: Boundary Equilibrium Generative Adversarial Networks. arXiv:1703.10717 [cs.LG]

Strebelle S., 2002. Conditional Simulation of Complex Geological Structures Using Multiple-Point Statistics. Mathematical Geology, 34, 1-21. https://doi.org/10.1023/A:1014009426274

Emerick A.A., Reynolds A.C., 2013. Ensemble Smoother with multiple data assimilation. Computers & Geosciences, 55, 3-15. https://doi.org/10.1016/j.cageo.2012.03.011

Xie Y., Lin T., Chen Z., Xiong W., Ran Q., Shang C., 2022. A lightweight ensemble discriminator for Generative Adversarial Networks. Knowledge-Based Systems (in press). https://doi.org/10.1016/j.knosys.2022.108975

Conclusions

Thank you

for your attention!

ranazzi@usp.br

Appendix: Structures

- Networks Architecture:

DCGAN/WGAN

Generator	Config. - Output - Activ.
Input	[500]
Fully-conected	7x7x8n
Reshape	[7x7x8n]
2D conv. Transpose	filters = 8n, size=(5,5), strides=(2,2), padding=same - [14x14x128] - ReLU
Resize	Bilinear - [13x13x2n]
2D conv. Transpose	filters = 4n, size=(5,5), strides=(2,2), padding=same - [26x26x64] - ReLU
2D conv. Transpose	filters = 2n, size=(5,5), strides=(2,2), padding=same - [52x52x32] - ReLU
Resize	Bilinear - [51x51xn]
2D conv. Transpose	filters = 2, size=(5,5), strides=(1,1), padding=same - [51x51x2] - tanh

Discriminator	Config. - Output - Activ.
Input	[51x51x2]
2D conv.	filters = n, size=(4,4), strides=(2,2), padding=same - [26x26x32] - LeakyReLU
2D conv.	filters = 2n, size=(4,4), strides=(2,2), padding=same - [13x13x64] - LeakyReLU
2D conv.	filters = 4n, size=(4,4), strides=(2,2), padding=same - [7x7x128] - LeakyReLU
2D conv.	filters = 8n, size=(4,4), strides=(1,1), padding=same - [7x7x256] - LeakyReLU
Flatten	-
Fully-conected	1 - sigmoid/linear

BEGAN

Encoder	Config. - Output - Activ.
Input	[51x51x2]
2D conv.	filters = n, size=(3,3), strides=(1,1), padding=same - [51x51x32] - LeakyReLU
2D conv.	filters = n, size=(3,3), strides=(2,2), padding=same - [26x26x32] - LeakyReLU
2D conv.	filters = 2n, size=(3,3), strides=(1,1), padding=same - [26x26x64] - LeakyReLU
2D conv.	filters = 2n, size=(3,3), strides=(2,2), padding=same - [13x13x64] - LeakyReLU
2D conv.	filters = 3n, size=(3,3), strides=(1,1), padding=same - [13x13x96] - LeakyReLU
2D conv.	filters = 3n, size=(3,3), strides=(2,2), padding=same - [7x7x96] - LeakyReLU
2D conv.	filters = 4n, size=(3,3), strides=(1,1), padding=same - [7x7x128] - LeakyReLU
2D conv.	filters = 4n, size=(3,3), strides=(1,1), padding=same - [7x7x128] - LeakyReLU
Flatten	-
Fully-conected	[500] - tanh

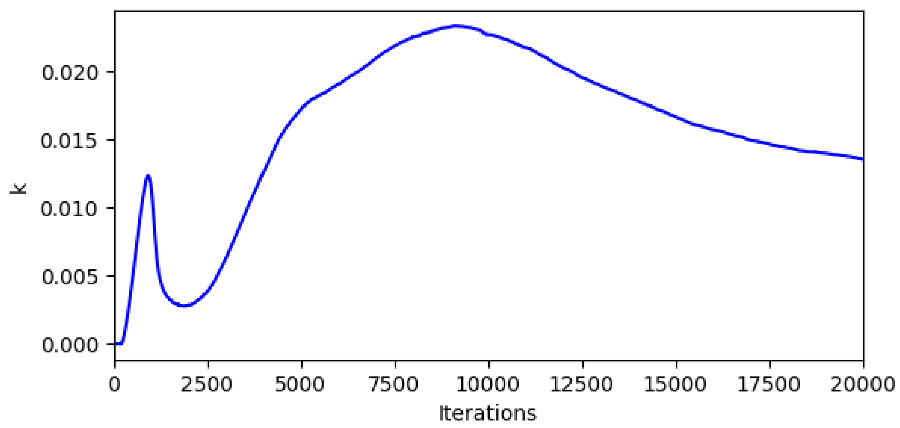
Decoder/Generator	Config. - Output - Activ.
Input	[500]
Fully-conected	7x7xn
Reshape	[7x7xn]
2D conv.	filters = n, size=(3,3), strides=(1,1), padding=same - [7x7x32] - LeakyReLU
Resize 1	Nearest Neighbour - [13x13x32]
Skip Connection 1	Concatenate [Resize1, Reshape]
2D conv.	filters = n, size=(3,3), strides=(1,1), padding=same - [13x13x32] - LeakyReLU
Resize 2	Nearest Neighbour - [26x26x32]
Skip Connection 2	Concatenate [Resize2, Reshape]
2D conv.	filters = n, size=(3,3), strides=(1,1), padding=same - [26x26x32] - LeakyReLU
Resize 3	Nearest Neighbour - [51x51x32]
2D conv.	filters = n, size=(3,3), strides=(1,1), padding=same - [51x51x32] - LeakyReLU
2D conv.	filters = 2, size=(3,3), strides=(1,1), padding=same - [51x51x2] - tanh

Appendix: Hyperparameters

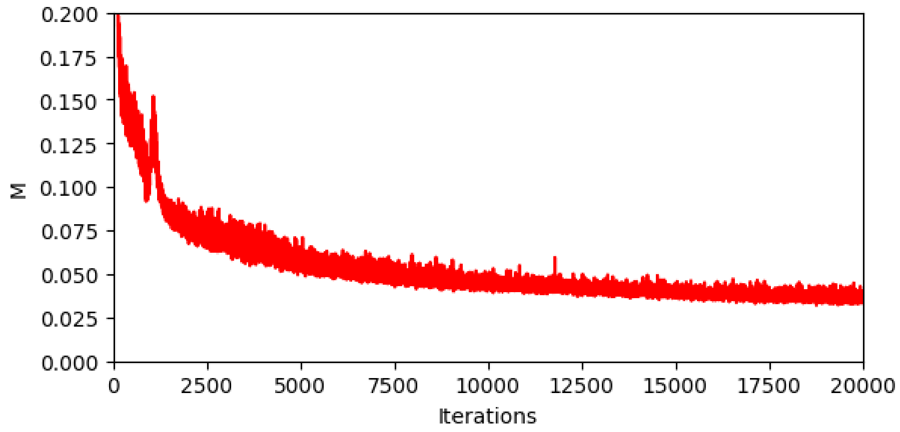
- Networks hyperparameters:

Hyperparameters		
GAN	**WGAN**	**BEGAN**
$N_z = 500$		
Adam	RMSprop	Adam
$\alpha_{\mathcal{G}} = \alpha_{\mathcal{D}} = 0.0002$*	$\alpha_{\mathcal{G}} = \alpha_{\mathcal{D}} = 1E-5$	$\alpha_{\mathcal{G}} = \alpha_{\mathcal{D}} = 0.0001$**
	$n_{critic} = 5$	$h = 500$
	$c = 0.05$	$\lambda_k = 0.001$
		$\gamma = 0.7$
* Generator trained twice in relation to discriminator ** Exponential decay at 10000 steps with rate equal to 0.5		

Appendix: Losses



κ_t



Convergence measure $\mathcal{M}$