▶ Notebook

Lecture 3 at a glance: six parts, seventeen sections

Part 1 · A learning signal

1.1 A generator needs feedback ▶

1.2 Two players ▶

Part 2 · One discriminator update

2.1 Data, labels and two small networks ▶

2.2 The discriminator's loss ▶

2.3 One D update, then check ▶

2.4 Detach the fake ▶

Part 3 · One generator update

3.1 The generator's goal ▶

3.2 One G update, then check ▶

3.3 Freeze versus detach ▶

Part 4 · Alternate the players

4.1 The training loop ▶

4.2 Minimax and saturation ▶

4.3 One neural update ▶

Part 5 · Samples, coverage and failure

5.1 Coverage and collapse ▶

5.2 What a loss curve cannot tell ▶

Part 6 · MNIST and retrieval

6.1 Images in and out ▶

6.2 Training on MNIST ▶

6.3 Exit and next ▶

> Six parts, seventeen numbered sections. The ▶ links open the matching notebook cell in Colab.

▶ Notebook

Part 1 · A learning signal

1 · A learning signal | 2 · One discriminator update | 3 · One generator update | 4 · Alternate the players | 5 · Samples, coverage and failure | 6 · MNIST and retrieval

A generator can sample, but it cannot score its own samples. What tells it how to improve?

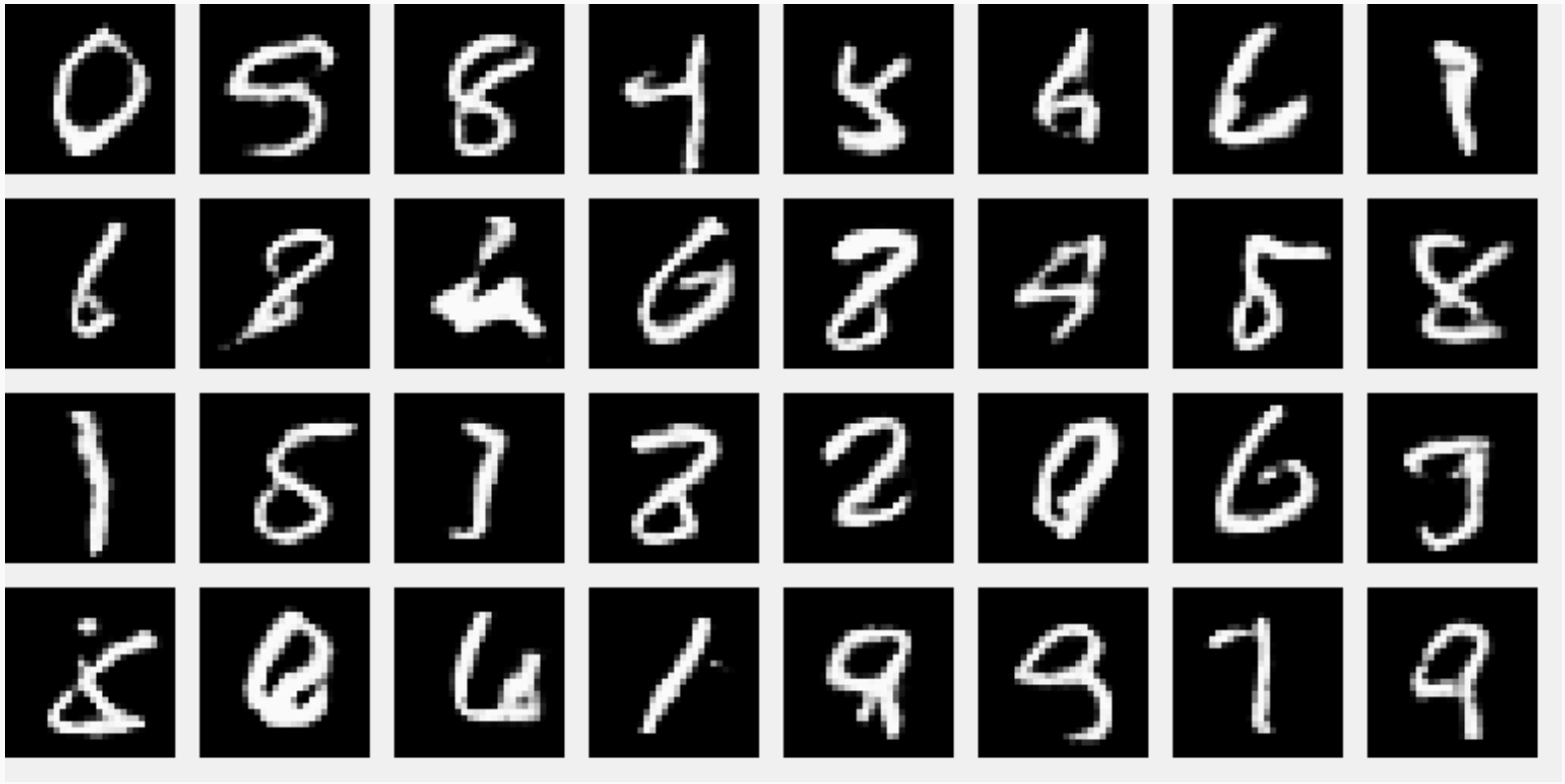
1.1 A generator needs feedback ▸

1.2 Two players ▸

> Part 1 of 6: two sections. Each ▶ opens the notebook cell that runs it.

Notebook

Where could a generator get feedback?



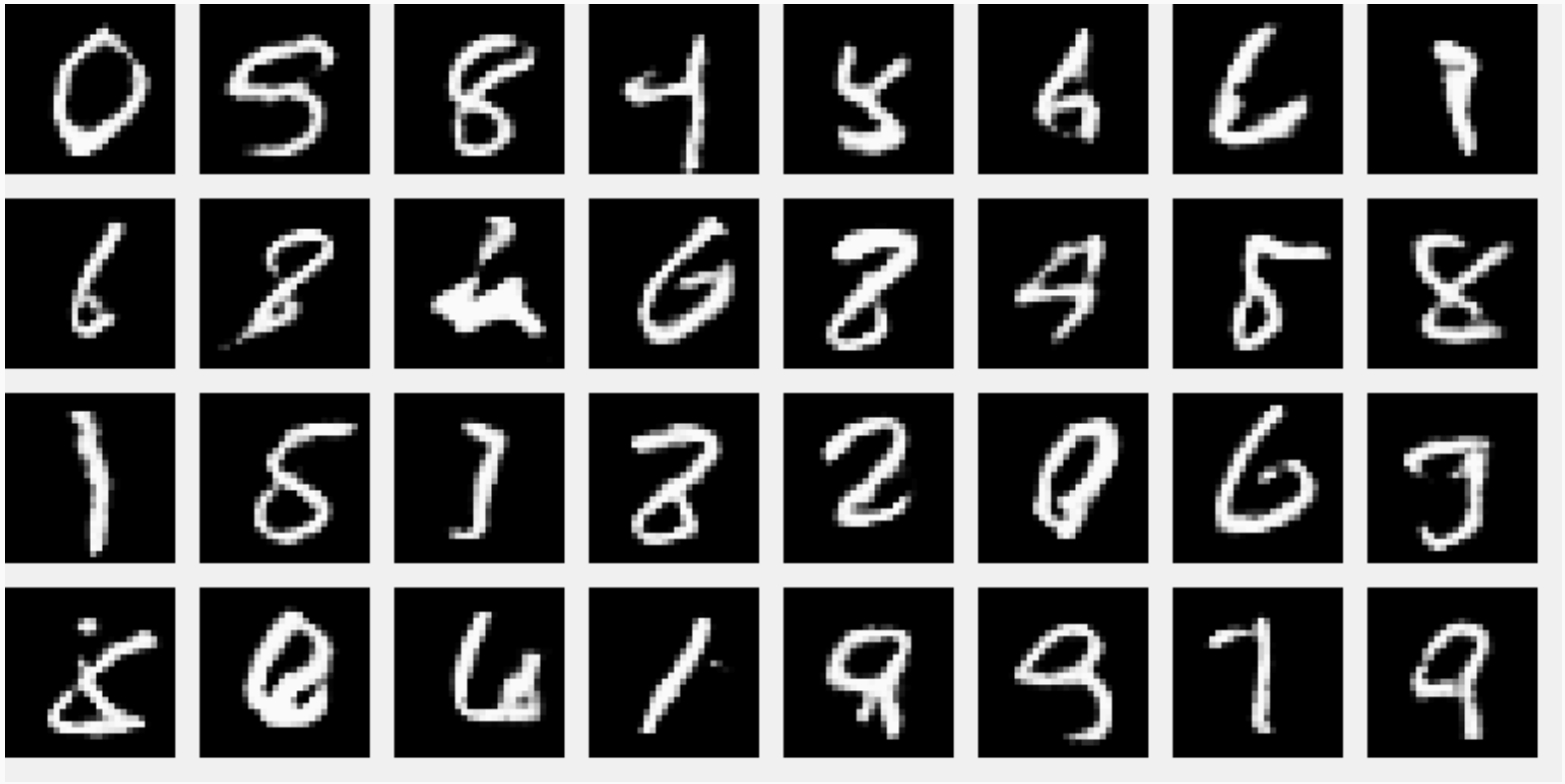
Measured replay · 30-epoch MNIST GAN · fixed noise seed 123

> What training signal could produce these samples?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Where could a generator get feedback?



Measured replay · 30-epoch MNIST GAN · fixed noise seed 123

We can sample from this network. What tells it how to improve?

> A learned discriminator can supply differentiable feedback.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A generator turns a draw into a sample



Which part introduces randomness?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A generator turns a draw into a sample



$$z \sim p(z), \qquad \tilde{x} = G_\theta(z)$$

> The sampled z; G is deterministic once z and θ are fixed.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Sampling is easier than evaluating a density

$$z \longmapsto G_\theta(z)$$

> Can this network readily report the probability of an image?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Sampling is easier than evaluating a density

$$z \longmapsto G_\theta(z)$$

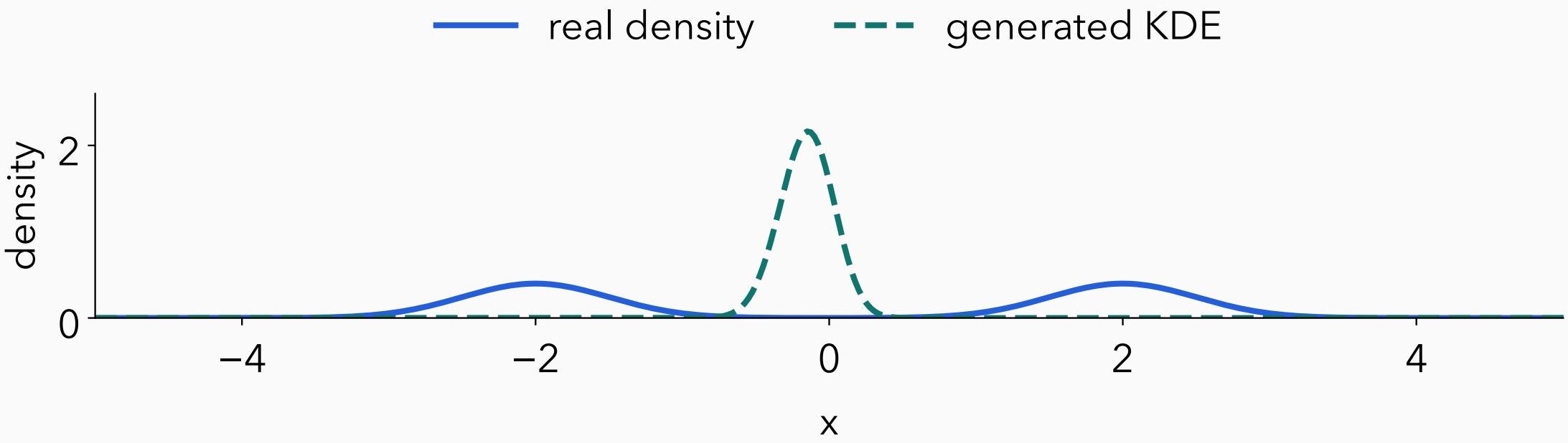
We do not need to evaluate $p_\theta(x)$ to perform the GAN updates.

> The usual GAN supplies an implicit sampler; its training signal comes from a classifier.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

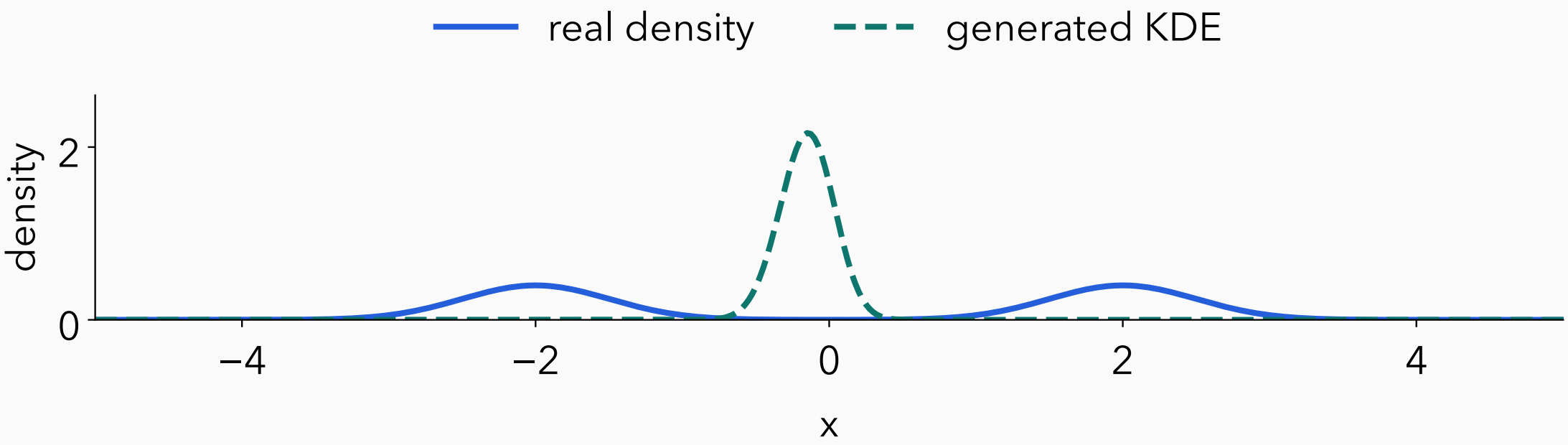
Which distribution is missing something?



Where should generated samples move?

▶ Notebook

Which distribution is missing something?

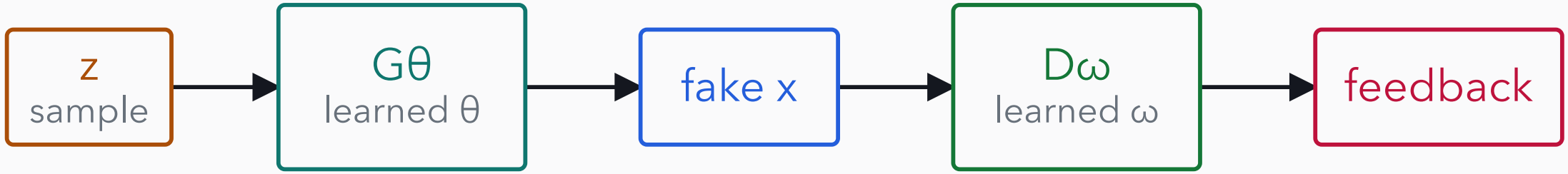


Known data: equal mixture at −2 and +2, standard deviation 0.5. KDE is a visual summary of generated samples.

> They need to cover both observed regions, not merely one plausible point.

▶ Notebook

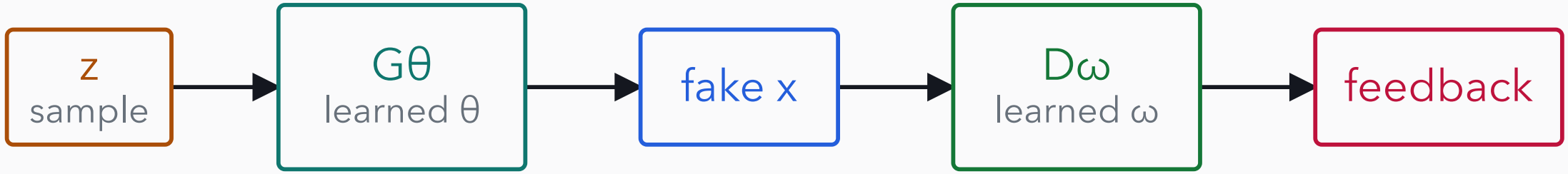
One network proposes; one distinguishes



> Must we keep D to generate after training?

▶ Notebook

One network proposes; one distinguishes



The discriminator is used during training. Sampling later needs only G.

> No. Draw z and run G.

▶ Notebook

Part 2 · One discriminator update

1 · A learning signal | 2 · One discriminator update | 3 · One generator update | 4 · Alternate the players | 5 · Samples, coverage and failure | 6 · MNIST and retrieval

Train a classifier to tell real from generated, one inspected step at a time.

2.1 Data, labels and two small networks ▶

2.2 The discriminator's loss ▶

2.3 One D update, then check ▶

2.4 Detach the fake ▶

> Part 2 of 6: four sections. Each ▶ opens the notebook cell that runs it.

▶ Notebook

Fix the samples and label their source

example	source	target y
x = 2	observed	1
G(1) = 0	generated	0

> Does label 1 mean “a perfect digit”?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Fix the samples and label their source

example	source	target y
x = 2	observed	1
G(1) = 0	generated	0

$$D_\omega(x) = \text{sigmoid}(wx + c)$$

> It means the example came from the observed-data source in this classification task.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Choose networks small enough to inspect

$$G_\theta(z) = az + b$$
$$a = 0.5,\ b = -0.5,\ z = 1$$

$$D_\omega(x) = \sigma(wx + c)$$
$$w = 0.4,\ c = -0.2$$

> What are the trainable quantities?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Choose networks small enough to inspect

$$G_\theta(z) = az + b$$
$$a = 0.5,\ b = -0.5,\ z = 1$$

$$D_\omega(x) = \sigma(wx + c)$$
$$w = 0.4,\ c = -0.2$$

Chosen scalar teaching example. θ = (a,b), ω = (w,c).

> a,b belong to G; w,c belong to D. x and z are fixed for this calculation.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

First compute the fake

$$\widetilde{x} = 0.5 \times 1 - 0.5$$

> What sample does this z produce?

▶ Notebook

First compute the fake

$$\widetilde{x} = 0.5 \times 1 - 0.5$$

$$\widetilde{x} = 0$$

> Zero. The observed example is at x = 2.

▶ Notebook

A score comes before a probability

input	logit wx+c	sigmoid
real x = 2	0.6	0.6457
fake x = 0	−0.2	0.4502

> Why keep raw logits in the implementation?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A score comes before a probability

input	logit wx+c	sigmoid
real x = 2	0.6	0.6457
fake x = 0	−0.2	0.4502

A logit is any real number. D(x) is between 0 and 1.

> Stable binary cross-entropy can operate directly on logits.

▶ Notebook

Reward a high score on real data

$$\ell_{\text{real}} = -\log D_\omega(2)$$

> Would D(real) closer to one increase this loss?

▶ Notebook

Reward a high score on real data

$$\ell_{\text{real}} = -\log D_\omega(2)$$

$$\ell_{\text{real}} \approx -\log 0.6457 \approx 0.4375$$

No, it reduces the negative log probability of the correct label.

▶ Notebook

Reward a low score on generated data

$$\ell_{\text{fake}} = -\log\big(1 - D_\omega(0)\big)$$

Which direction should D(fake) move?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Reward a low score on generated data

$$\ell_{\text{fake}} = -\log\big(1 - D_\omega(0)\big)$$

$$\ell_{\text{fake}} \approx -\log(1 - 0.4502) \approx 0.5981$$

> Towards zero during the discriminator phase.

▶ Notebook

Add the two classification losses

$$L_D = -\log D_\omega(x) - \log(1 - D_\omega(G_\theta(z)))$$

One real + one fake. For batches: mean over real examples + mean over fake examples.

> Is this an average over the combined two-example batch?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Add the two classification losses

$$L_D = -\log D_\omega(x) - \log(1 - D_\omega(G_\theta(z)))$$

$$L_D \approx 1.0356$$

One real + one fake. For batches: mean over real examples + mean over fake examples.

> Our convention sums the two source means; a combined mean differs by a factor of two.

▶ Notebook

Which derivative changes the classifier?

$$\frac{\partial L_D}{\partial w} = (D(x) - 1)x + D(\tilde{x})\tilde{x}$$

> Which term contributes to the weight derivative when fake x is zero?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Which derivative changes the classifier?

$$\frac{\partial L_D}{\partial w} = (D(x) - 1)x + D(\tilde{x})\tilde{x}$$

$$\frac{\partial L_D}{\partial w} \approx -0.7087, \quad \frac{\partial L_D}{\partial c} \approx 0.0958$$

Only the real term contributes to ∂L/∂w; both contribute to the bias derivative.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

One D update, with G unchanged

parameter	before	after η = 0.1
w	0.4000	0.4709
c	-0.2000	-0.2096
a, b	0.5, −0.5	0.5, −0.5

> Which row must remain identical?

▶ Notebook

One D update, with G unchanged

parameter	before	after η = 0.1
w	0.4000	0.4709
c	-0.2000	-0.2096
a, b	0.5, −0.5	0.5, −0.5

$$\omega^{+} = \omega - 0.1\nabla_{\omega} L_D$$

> The generator parameters. This phase updates only ω.

▶ Notebook

Check the same examples after the update

quantity	before	after
D(real)	0.6457	0.6753
D(fake)	0.4502	0.4478
D loss	1.0356	0.9865

> Did the discriminator improve on these two examples?

▶ Notebook

Check the same examples after the update

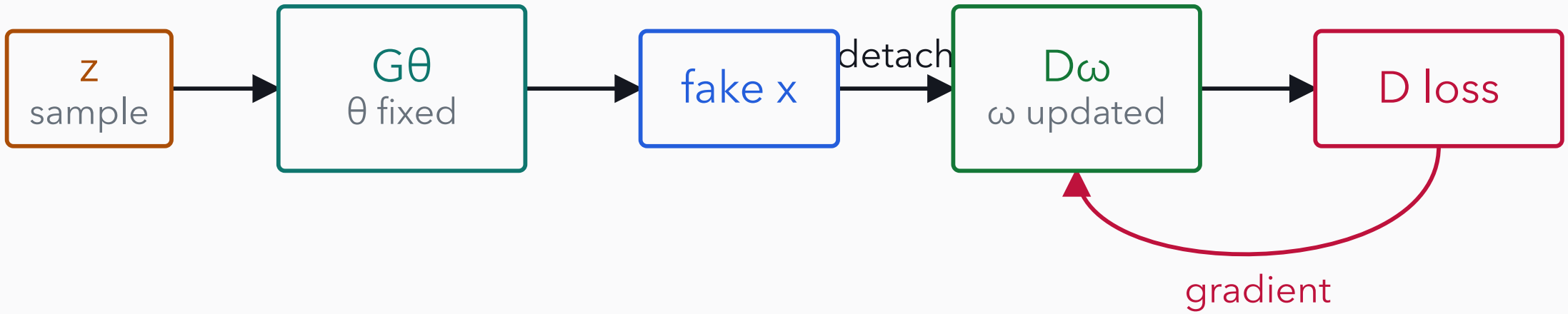
quantity	before	after
D(real)	0.6457	0.6753
D(fake)	0.4502	0.4478
D loss	1.0356	0.9865

Computed on the same fixed x and z. This checks one update, not generalisation.

> Yes: real score rises, fake score falls, and the same-batch loss falls.

▶ Notebook

Stop this phase at the fake sample

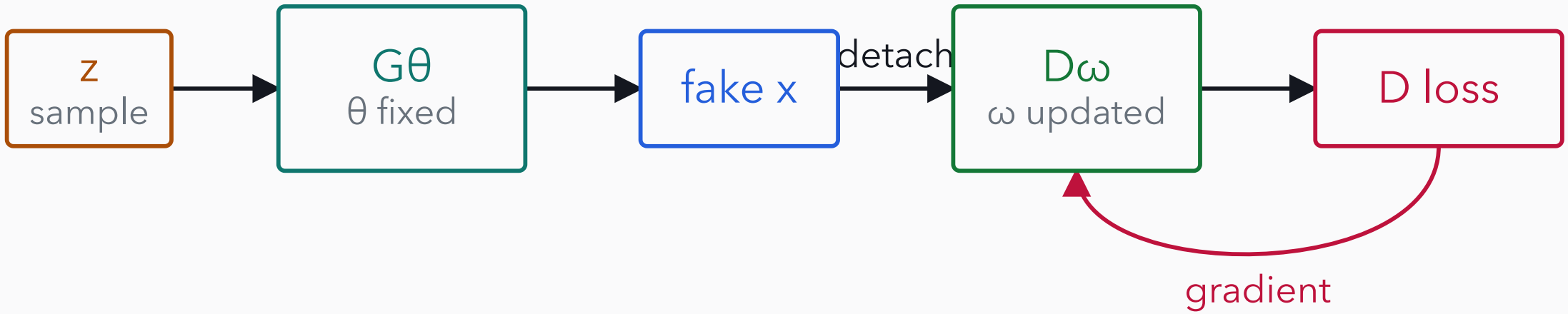


> Why detach fake rather than remove D from the graph?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Stop this phase at the fake sample



The coral derivative stops at D. No derivative from this phase enters G.

> D needs gradients; G should not receive this phase's gradients.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Detach only the generated sample

```
fake = G(z)
fake_logits = D(fake.detach())
loss_D.backward()
opt_D.step()
```

> Which part of this computation must still be differentiable?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Detach only the generated sample

```
fake = G(z)
fake_logits = D(fake.detach())
loss_D.backward()
opt_D.step()
```

The real-data branch contributes to the same D loss.

> The discriminator forward pass; only the path back into G is cut.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Check ownership of the update

D phase assertion	observed
G parameter change	exactly zero
G gradient from this loss	None
D parameter change	nonzero

> Does a nonzero stale .grad prove this phase updated a parameter?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Check ownership of the update

D phase assertion	observed
G parameter change	exactly zero
G gradient from this loss	None
D parameter change	nonzero

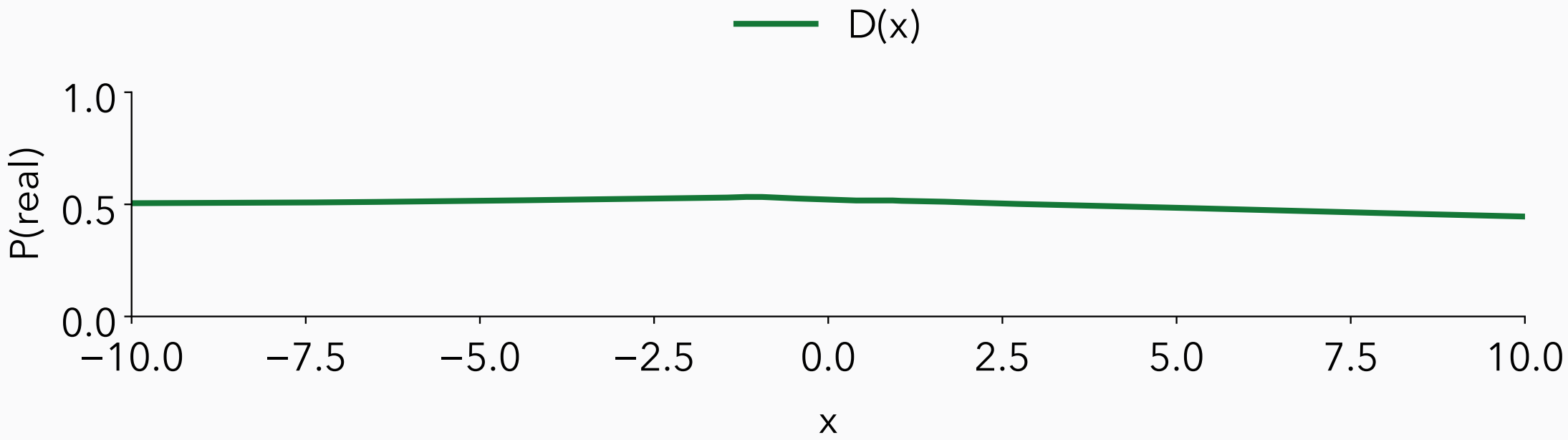
Verified by the executable float64 scalar example; no previous gradients are left in the test.

> No. Clear gradients and compare parameter values before and after the actual optimiser step.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

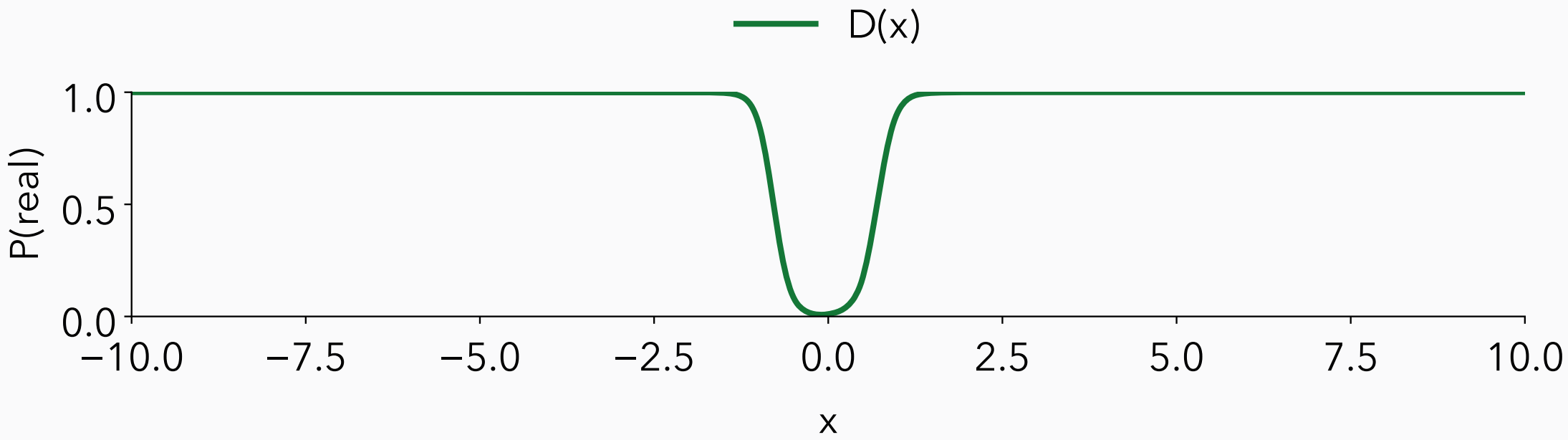
What happens if only D learns?



Predict the score near the real modes and the fake region.

▶ Notebook

What happens if only D learns?



Measured saved run · seed 4 · G fixed at initialisation

> D learns to distinguish these fixed sources; the generator itself has not improved.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Part 3 · One generator update

1 · A learning signal | 2 · One discriminator update | 3 · One generator update | 4 · Alternate the players | 5 · Samples, coverage and failure | 6 · MNIST and retrieval

Now the generator asks for the label “real”, through a frozen critic.

3.1 The generator’s goal ▶

3.2 One G update, then check ▶

3.3 Freeze versus detach ▶

> Part 3 of 6: three sections. Each ▶ opens the notebook cell that runs it.

▶ Notebook

Now let the generator ask for label 1

$$L_G = -\log D_\omega(G_\theta(z))$$

> Are we relabelling generated data for the next D update?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Now let the generator ask for label 1

$$L_G = -\log D_\omega(G_\theta(z))$$

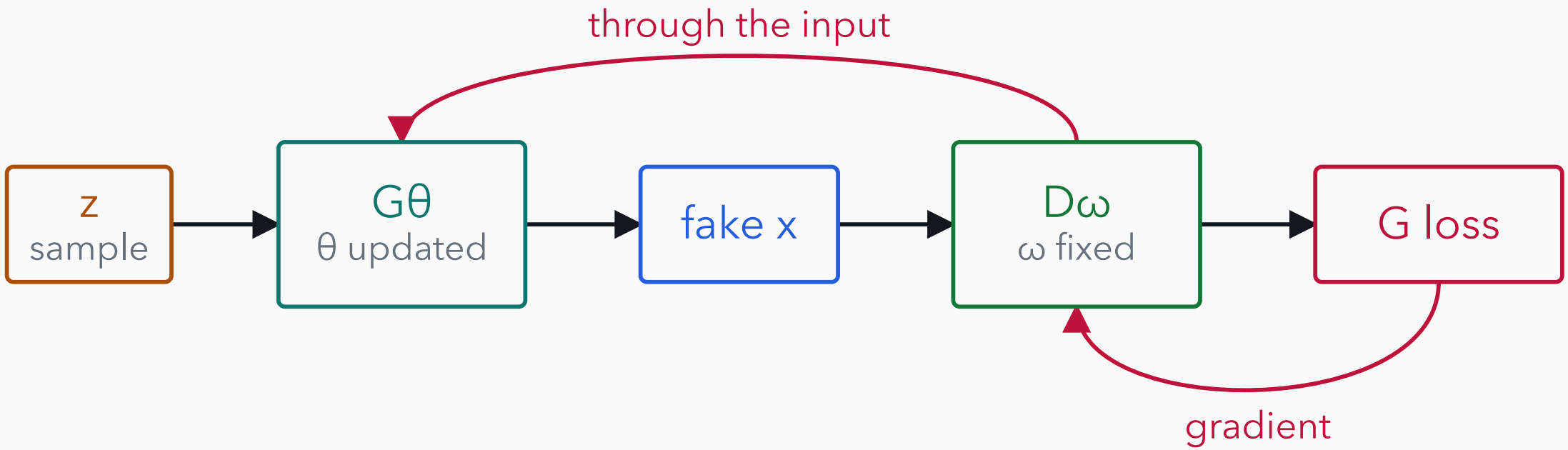
For the G update, a generated sample should receive a high real-source score.

> No. Label 1 defines G's objective; D still trains fake examples with label 0.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

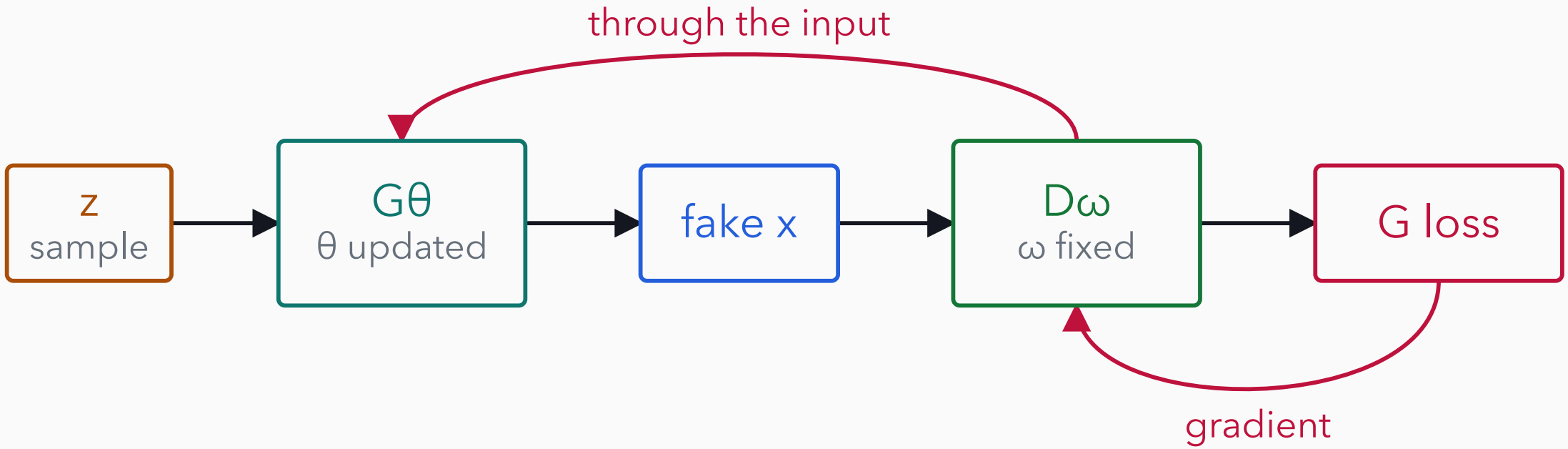
Frozen weights can still transmit a gradient



What does "freeze D" mean here?

▶ Notebook

Frozen weights can still transmit a gradient



ω is fixed; the derivative of D with respect to its input is still needed.

> Do not update ω. Preserve the differentiable path from loss through D to G.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Follow the chain rule into the generator

$$\frac{\partial L_G}{\partial a} = \frac{\partial L_G}{\partial s}\frac{\partial s}{\partial \tilde{x}}\frac{\partial \tilde{x}}{\partial a}$$

s = w·fake + c is the discriminator logit. ω is fixed.

> Can a fixed w appear in a derivative?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Follow the chain rule into the generator

$$\frac{\partial L_G}{\partial a} = \frac{\partial L_G}{\partial s}\frac{\partial s}{\partial \tilde{x}}\frac{\partial \tilde{x}}{\partial a}$$

$$= (D(\tilde{x}) - 1)\, w\, z$$

s = w·fake + c is the discriminator logit. ω is fixed.

> Yes. It multiplies the input derivative even though we do not optimise w in this phase.

Notebook

The generator receives a nonzero derivative

$$\frac{\partial L_G}{\partial a} \approx (0.4478 - 1) \times 0.4709 \times 1$$

> Will gradient descent move a up or down?

Notebook

The generator receives a nonzero derivative

$$\frac{\partial L_G}{\partial a} \approx (0.4478 - 1) \times 0.4709 \times 1$$

$$\frac{\partial L_G}{\partial a} \approx -0.2600, \quad \frac{\partial L_G}{\partial b} \approx -0.2600$$

Up: subtracting a negative derivative increases a.

▶ Notebook

One G update, with D unchanged

parameter	before	after η = 0.1
a	0.5000	0.5260
b	−0.5000	-0.4740
w, c	0.4709, -0.2096	0.4709, -0.2096

> Which object moved even though the random draw stayed fixed?

▶ Notebook

One G update, with D unchanged

parameter	before	after η = 0.1
a	0.5000	0.5260
b	−0.5000	-0.4740
w, c	0.4709, -0.2096	0.4709, -0.2096

$$\tilde{x}^{+} = a^{+} z + b^{+} \approx 0.0520$$

> The generated sample moved because G's parameters changed.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Check the updated sample and loss

quantity	before	after
fake sample	0.0000	0.0520
G loss	0.8034	0.7900
D parameter change	0	0

> Does this one improvement establish coverage of both modes?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Check the updated sample and loss

quantity	before	after
fake sample	0.0000	0.0520
G loss	0.8034	0.7900
D parameter change	0	0

Autodiff agrees with central finite differences to better than 10^{-9}.

> No. It checks the local learning signal for one fixed draw.

▶ Notebook

Keep the path; freeze the owner

```
D.requires_grad_(False)
loss_G = BCE_logits(D(G(z)), ones)
opt_G.zero_grad()
loss_G.backward()
opt_G.step()
D.requires_grad_(True)
```

> What would G(z).detach() do in this code?

▶ Notebook

Keep the path; freeze the owner

```
D.requires_grad_(False)
loss_G = BCE_logits(D(G(z)), ones)
opt_G.zero_grad()
loss_G.backward()
opt_G.step()
D.requires_grad_(True)
```

Use a fresh generator forward pass after the D update.

> It would sever the very path needed to update G.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Two operations with different effects

operation	D weight gradients	gradient into G
freeze D parameters	off	preserved
detach G(z)	may exist	cut
torch.no_grad() around D	off	cut

> Which operation is correct during the G phase?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Two operations with different effects

operation	D weight gradients	gradient into G
freeze D parameters	off	preserved
detach G(z)	may exist	cut
torch.no_grad() around D	off	cut

Freezing parameters is compatible with differentiating with respect to inputs.

> Freeze D parameters, keep ordinary autograd through its forward computation.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Verify the actual parameter changes

G phase assertion	observed
D parameters before versus after	identical
G gradients	nonzero
G parameters before versus after	changed

> Why compare parameters as well as gradients?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Verify the actual parameter changes

G phase assertion	observed
D parameters before versus after	identical
G gradients	nonzero
G parameters before versus after	changed

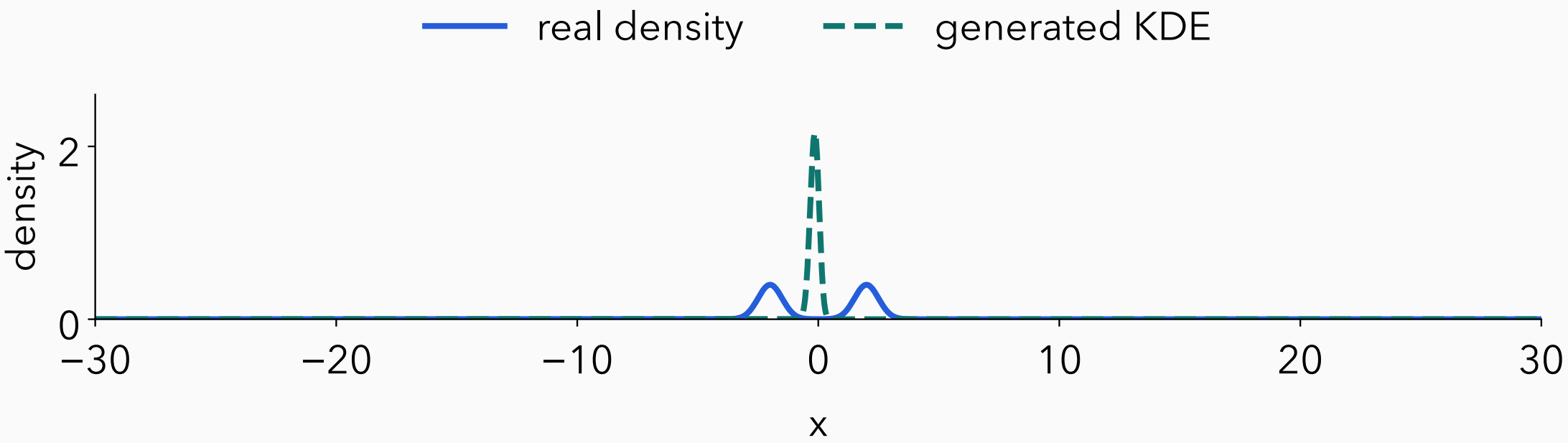
Notebook: scalar_updates() and the neural two-phase ownership test.

> The optimiser decides what actually changes; a gradient alone is insufficient evidence.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

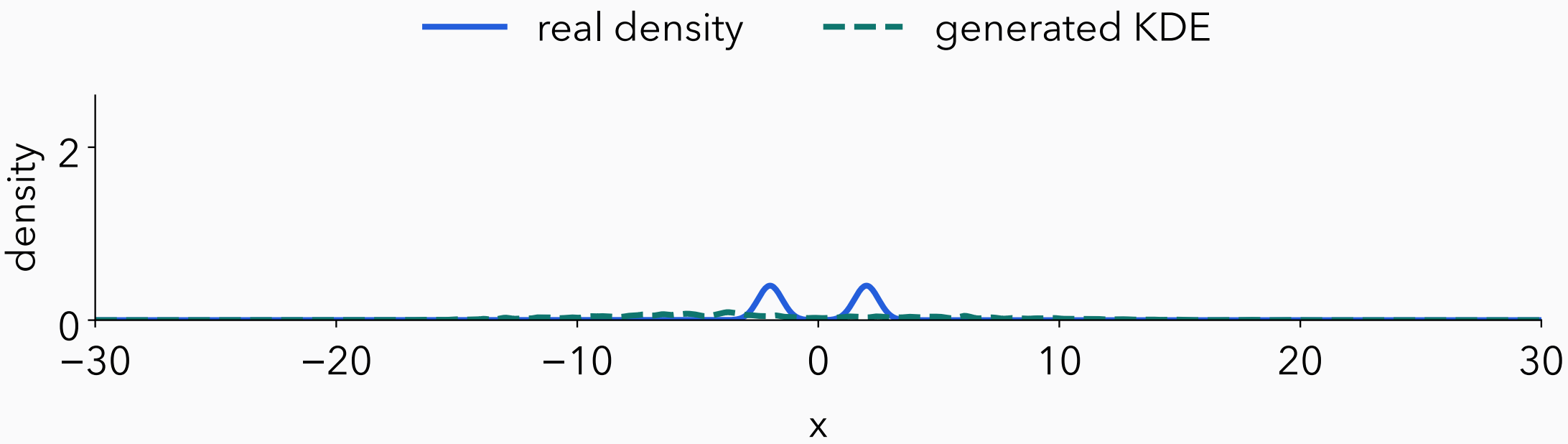
A fixed critic can be exploited



Would optimising against one fixed D guarantee the data distribution?

▶ Notebook

A fixed critic can be exploited



Measured saved run · fixed D · 56.6% outside [−5,5] · axes [−30,30]

> No. G can exploit that particular critic; alternating training changes the task.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Part 4 · Alternate the players

Alternate the two updates, and see why practice changes the generator's loss.

4.1 The training loop ▶

4.2 Minimax and saturation ▶

4.3 One neural update ▶

> Part 4 of 6: three sections. Each ▶ opens the notebook cell that runs it.

▶ Notebook

The training loop alternates two objectives

1 · Improve D on real and detached fake samples.

> Are both players minimising one fixed function of their own weights?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The training loop alternates two objectives

1 · Improve D on real and detached fake samples.

2 · Improve G through the newly updated, fixed D.

> Each player sees a changing opponent. The practical update uses two stated objectives.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The training loop alternates two objectives

1 · Improve D on real and detached fake samples.

$$\omega \leftarrow \omega - \eta_D \nabla_\omega L_D$$

2 · Improve G through the newly updated, fixed D.

$$\theta \leftarrow \theta - \eta_G \nabla_\theta L_G$$

> Each player sees a changing opponent. The practical update uses two stated objectives.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The discriminator has two expectations

$$L_D = -\mathbb{E}_{x \sim p_{\text{data}}} \log D_\omega(x)$$

> Where does each expectation get its samples?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The discriminator has two expectations

$$L_D = -\mathbb{E}_{x \sim p_{\text{data}}} \log D_\omega(x)$$

$$- \mathbb{E}_{z \sim p(z)} \log(1 - D_\omega(G_\theta(z)))$$

> Real examples come from the dataset; latent draws come from the chosen noise distribution.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The practical generator loss

$$L_G = -\mathbb{E}_{z\sim p(z)} \log D_\omega(G_\theta(z))$$

> Why is there no paired target image for each z?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The practical generator loss

$$L_G = -\mathbb{E}_{z\sim p(z)} \log D_\omega(G_\theta(z))$$

Non-saturating objective. Minimise over θ while holding ω fixed for this phase.

> G is judged by the discriminator, not by reconstruction of a paired observation.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The original game uses a different G term

$$\min_{\theta} \max_{\omega} \; \mathbb{E}_x \log D_\omega(x) + \mathbb{E}_z \log(1 - D_\omega(G_\theta(z)))$$

> Is −log D(fake) identical to log(1−D(fake))?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The original game uses a different G term

$$\min_{\theta} \max_{\omega} \; \mathbb{E}_x \log D_\omega(x) + \mathbb{E}_z \log(1 - D_\omega(G_\theta(z)))$$

The practical non-saturating G loss changes the training objective to obtain a stronger early signal.

> No. They encourage the same desired direction but have different gradients.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A confident discriminator can weaken one signal

G objective	derivative w.r.t. fake logit s
log(1−sigmoid(s))	−D(fake)
−log sigmoid(s)	D(fake)−1

> Which derivative is stronger when D rejects fake samples confidently?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A confident discriminator can weaken one signal

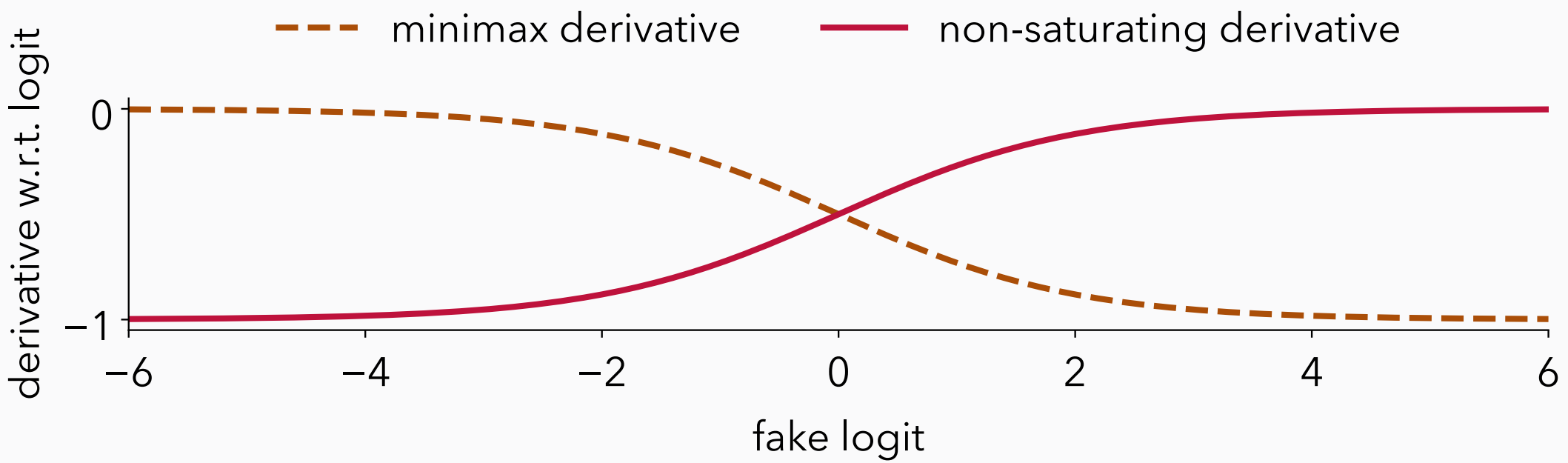
G objective	derivative w.r.t. fake logit s
log(1−sigmoid(s))	−D(fake)
−log sigmoid(s)	D(fake)−1

$$D(\tilde{x}) \approx 0: \qquad -D \approx 0, \quad D - 1 \approx -1$$

> The non-saturating objective's logit derivative is near −1.

▶ Notebook

The difference is visible before any training

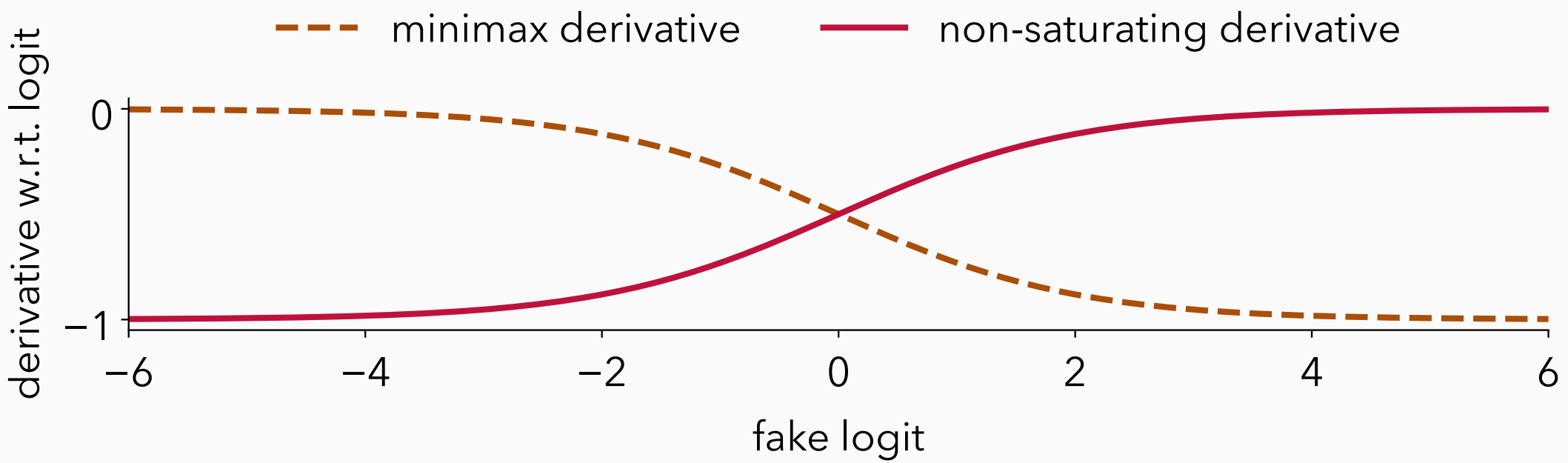


Does a stronger logit derivative guarantee stable GAN training?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The difference is visible before any training



This is a derivative with respect to the logit, not the complete gradient norm in G.

> No. The rest of the network Jacobian and the moving opponent still matter.

▶ Notebook

The first half of one neural update

```
fake = G(torch.randn(B, zdim))
loss_D = BCE_logits(D(real), ones)
loss_D += BCE_logits(D(fake.detach()), zeros)
opt_D.zero_grad(set_to_none=True)
loss_D.backward()
opt_D.step()
```

> Why detach only in this half?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The first half of one neural update

```
fake = G(torch.randn(B, zdim))
loss_D = BCE_logits(D(real), ones)
loss_D += BCE_logits(D(fake.detach()), zeros)
opt_D.zero_grad(set_to_none=True)
loss_D.backward()
opt_D.step()
```

Mean within each source, then sum the two source losses.

> This loss trains the discriminator while keeping G fixed.

▶ Notebook

The second half of one neural update

```
D.requires_grad_(False)
loss_G = BCE_logits(D(G(fresh_z)), ones)
opt_G.zero_grad(set_to_none=True)
loss_G.backward(); opt_G.step()
D.requires_grad_(True)
```

> What changed between the two forward passes?

▶ Notebook

The second half of one neural update

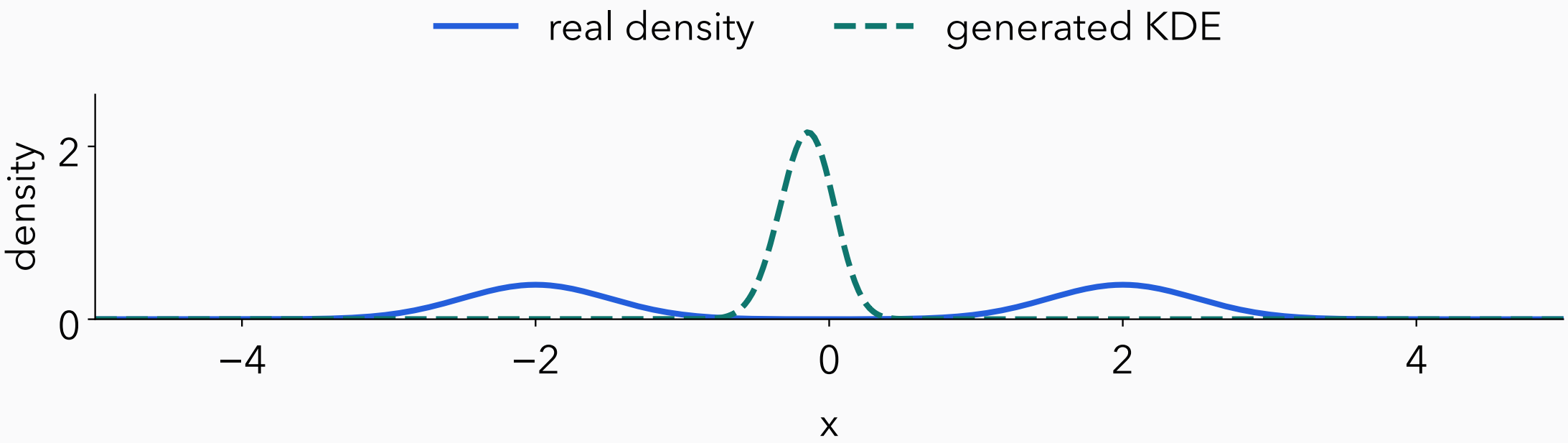
```
D.requires_grad_(False)
loss_G = BCE_logits(D(G(fresh_z)), ones)
opt_G.zero_grad(set_to_none=True)
loss_G.backward(); opt_G.step()
D.requires_grad_(True)
```

Do not reuse the discriminator's old forward graph after opt_D.step().

> D's parameters changed. G needs feedback from the updated discriminator.

▶ Notebook

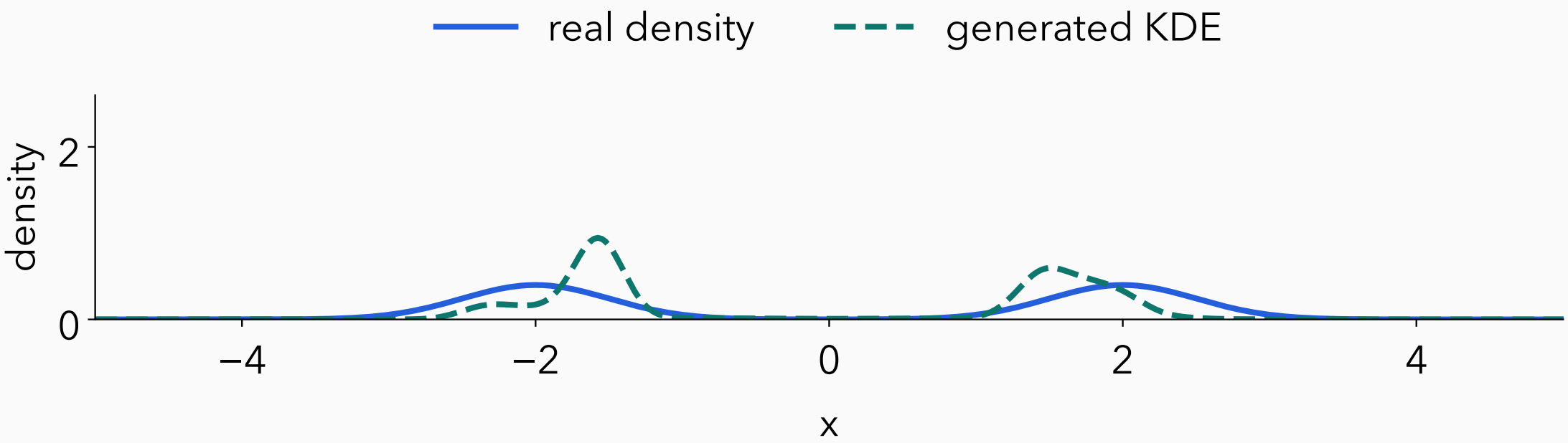
Can alternating updates cover both regions?



What would count as success in this simple test?

▶ Notebook

Can alternating updates cover both regions?



Saved run · seed 4 · G width 64 · 3,000 updates

> Samples near both modes, reasonable spread, and limited mass between them.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Part 5 · Samples, coverage and failure

A loss curve is not a quality certificate: count where the samples land.

5.1 Coverage and collapse ▶

5.2 What a loss curve cannot tell ▶

Part 5 of 6: two sections. Each ▶ opens the notebook cell that runs it.

▶ Notebook

Count mass near each known mode

region	wide generator	narrow generator
within 1 of −2	0.518	0.949
within 1 of +2	0.461	0.000

> What does a near-zero entry tell us?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Count mass near each known mode

region	wide generator	narrow generator
within 1 of −2	0.518	0.949
within 1 of +2	0.461	0.000

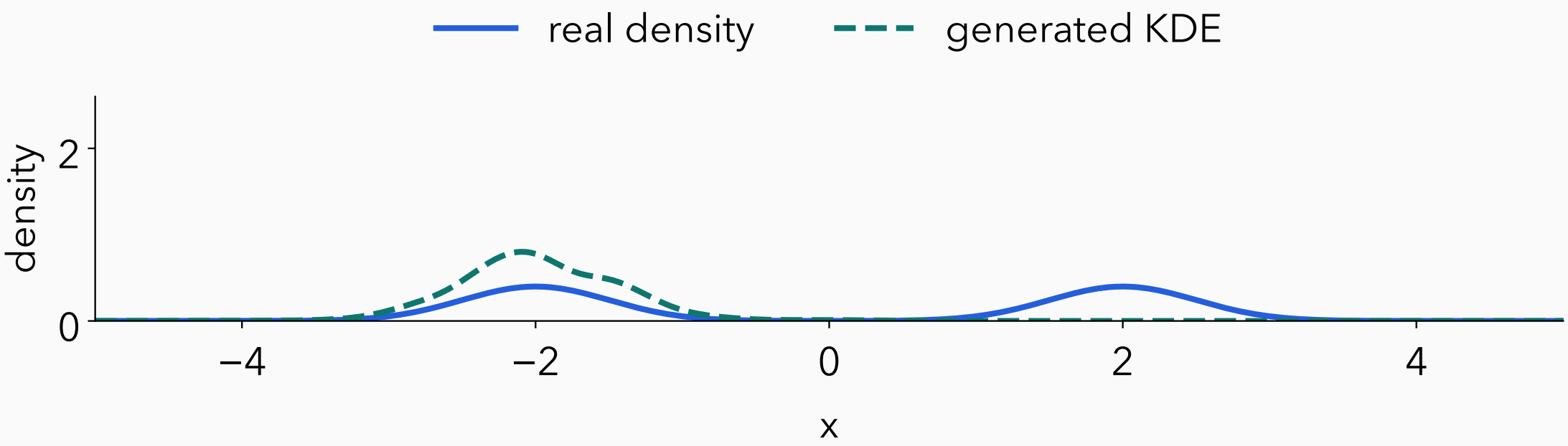
Measured fractions of 2,000 fixed-noise samples; intervals are chosen diagnostics, not a complete quality metric.

> The generator misses one observed region under this diagnostic.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Plausible samples can omit a whole mode



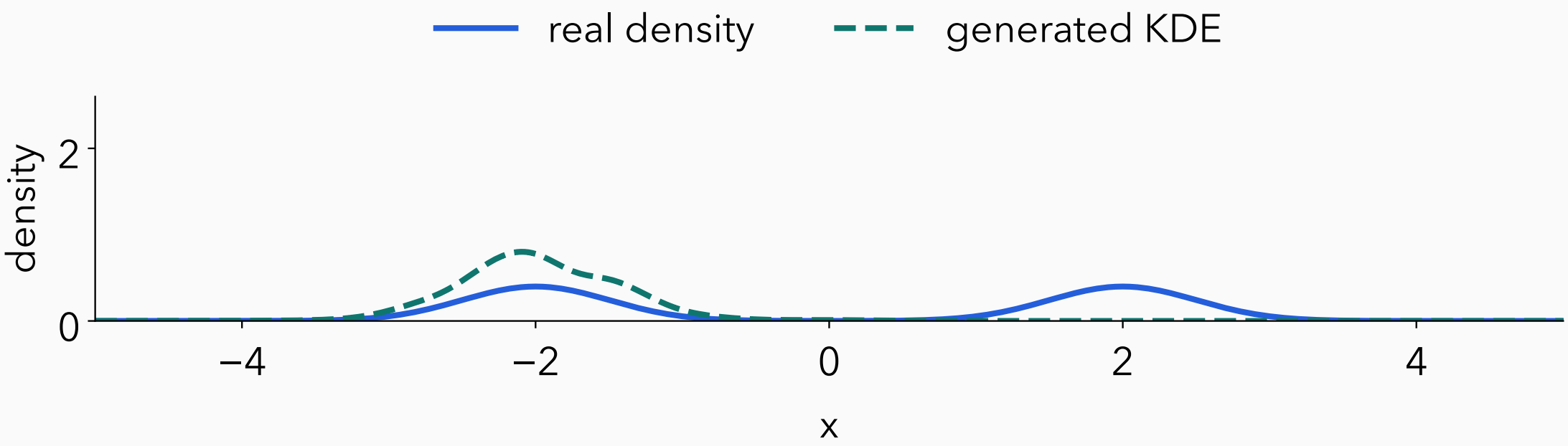
Measured run · seed 0 · G width 4 · 3,000 alternating updates

> Why is looking at a single generated sample insufficient?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Plausible samples can omit a whole mode



Measured run · seed 0 · G width 4 · 3,000 alternating updates

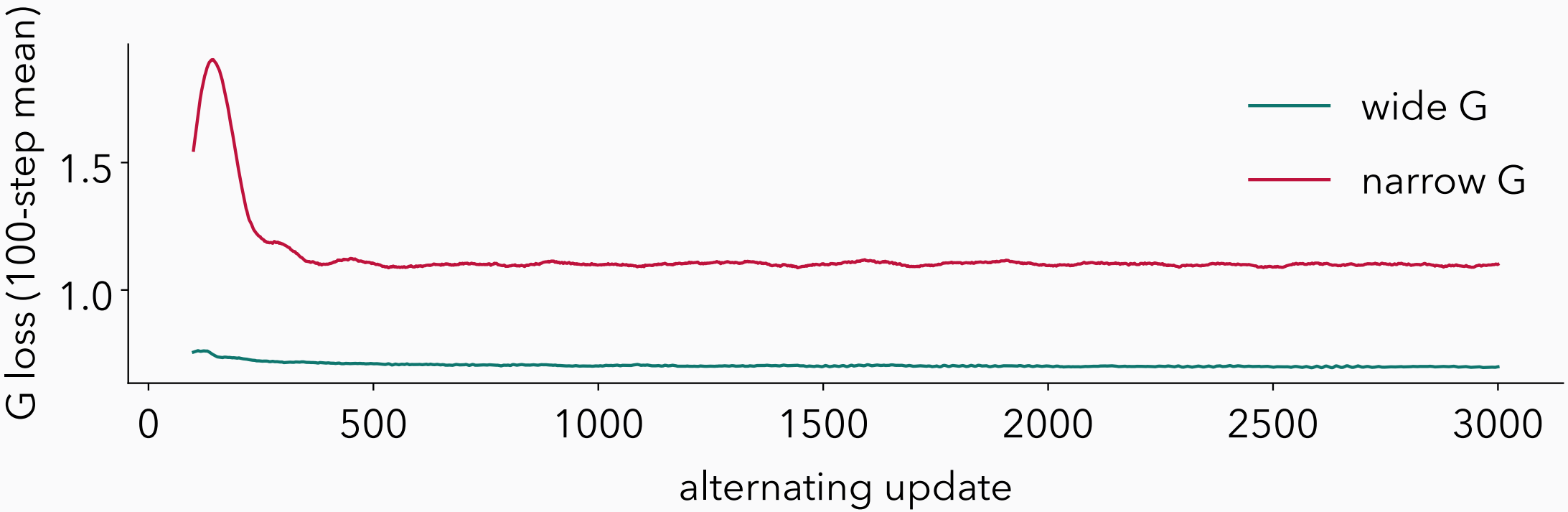
A sample can look plausible while the collection lacks diversity.

> It cannot reveal missing modes or the frequency of different outputs.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A loss curve is not a coverage certificate

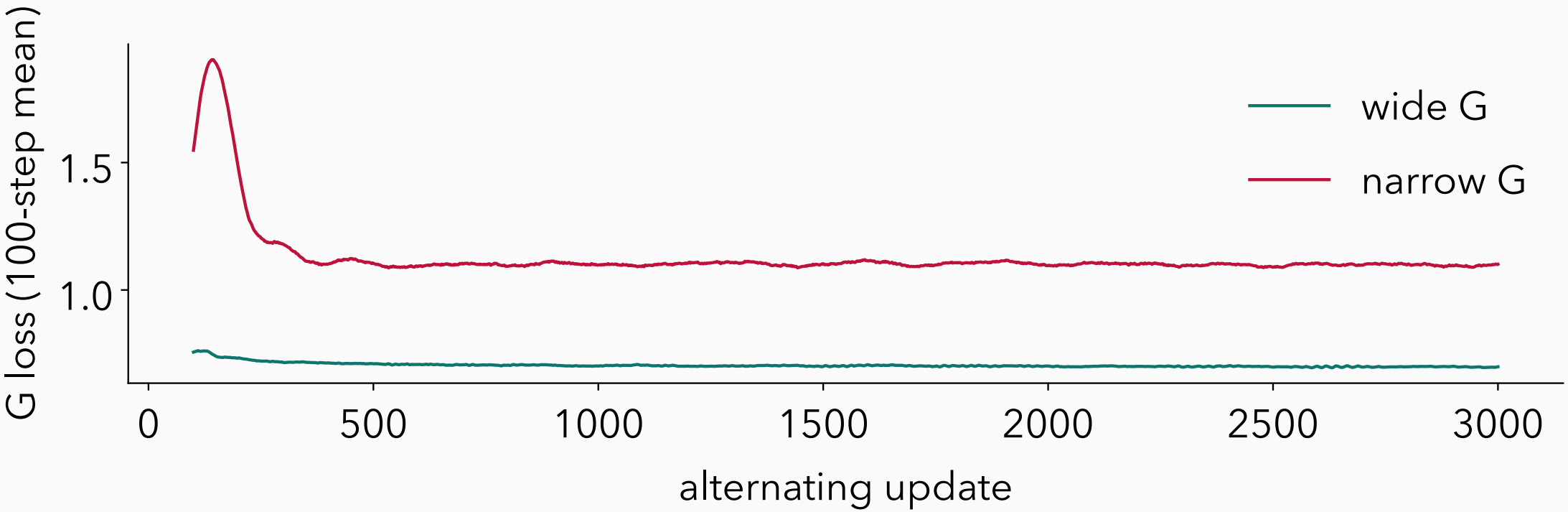


> Can the lower displayed G loss alone identify the better distribution?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

A loss curve is not a coverage certificate



Different opponents and model capacities. Interpret losses with samples and coverage checks.

> No. D changes during training, so the numerical loss is not a fixed external quality measure.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The target of optimisation keeps moving



> Must both plotted losses fall on every iteration?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The target of optimisation keeps moving



Progress for one player can make the other player's loss worse.

> No. The opponent changes the function being evaluated.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Ask separate questions about the samples

question	inspect
plausibility	fixed sample grid
coverage	many samples + class/mode counts
training changes	same latent draws over time
memorisation	held-out and nearest-neighbour checks

> What does fixed z control in a training film?

▶ Notebook

Ask separate questions about the samples

question	inspect
plausibility	fixed sample grid
coverage	many samples + class/mode counts
training changes	same latent draws over time
memorisation	held-out and nearest-neighbour checks

This package demonstrates sample/coverage checks; it does not claim a full memorisation or image-quality evaluation.

> It separates changes in G from changes in the sampled latent inputs.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Part 6 · MNIST and retrieval

1 · A learning signal | 2 · One discriminator update | 3 · One generator update | 4 · Alternate the players | 5 · Samples, coverage and failure | **6 · MNIST and retrieval**

The same game on images, then what we reproduced today.

6.1 Images in and out ▶

6.2 Training on MNIST ▶

6.3 Exit and next ▶

> Part 6 of 6: three sections. Each ▶ opens the notebook cell that runs it.

▶ Notebook

Keep the data convention visible



Greyscale MNIST

28 × 28 × 1

$x = \text{uint8}/127.5 - 1$

> Can we compare GAN loss numbers to the VAE reconstruction loss?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Keep the data convention visible



Greyscale MNIST

28 × 28 × 1

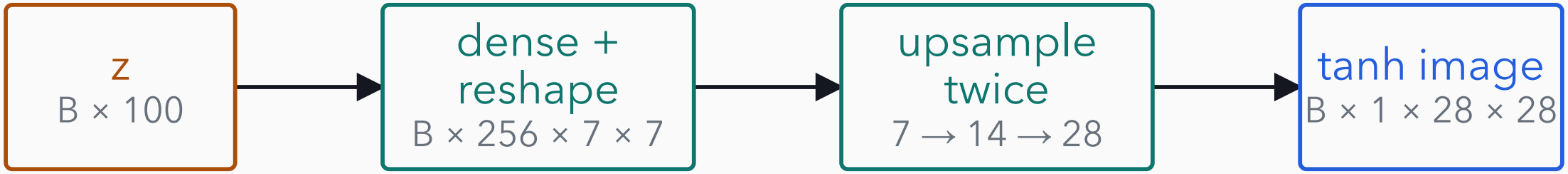
$$x = \text{uint8}/127.5 - 1$$

Same held 3 as earlier lectures. GAN pixels are greyscale in $[-1, 1]$; the Bernoulli VAE used binary observations.

> No. Their objectives, observation representations and reductions differ.

▶ Notebook

The generator expands a noise vector

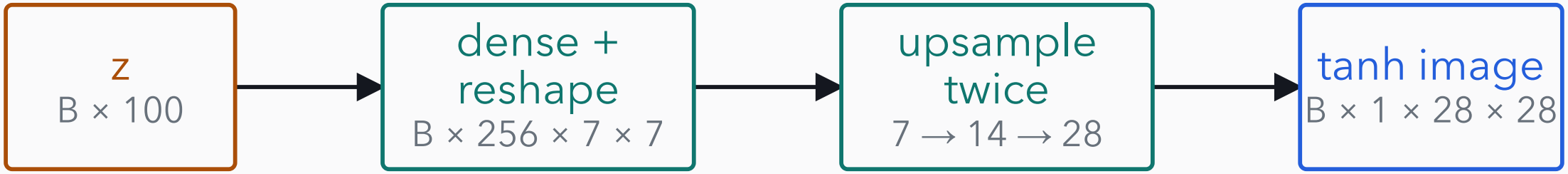


Why use tanh at the output here?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The generator expands a noise vector



Architecture of the saved checkpoint. The dense reshape does not imply independent output pixels.

> It matches the chosen $[-1,1]$ image representation.

▶ Notebook

The discriminator returns one logit per image



Does D predict one target per pixel?

▶ Notebook

The discriminator returns one logit per image

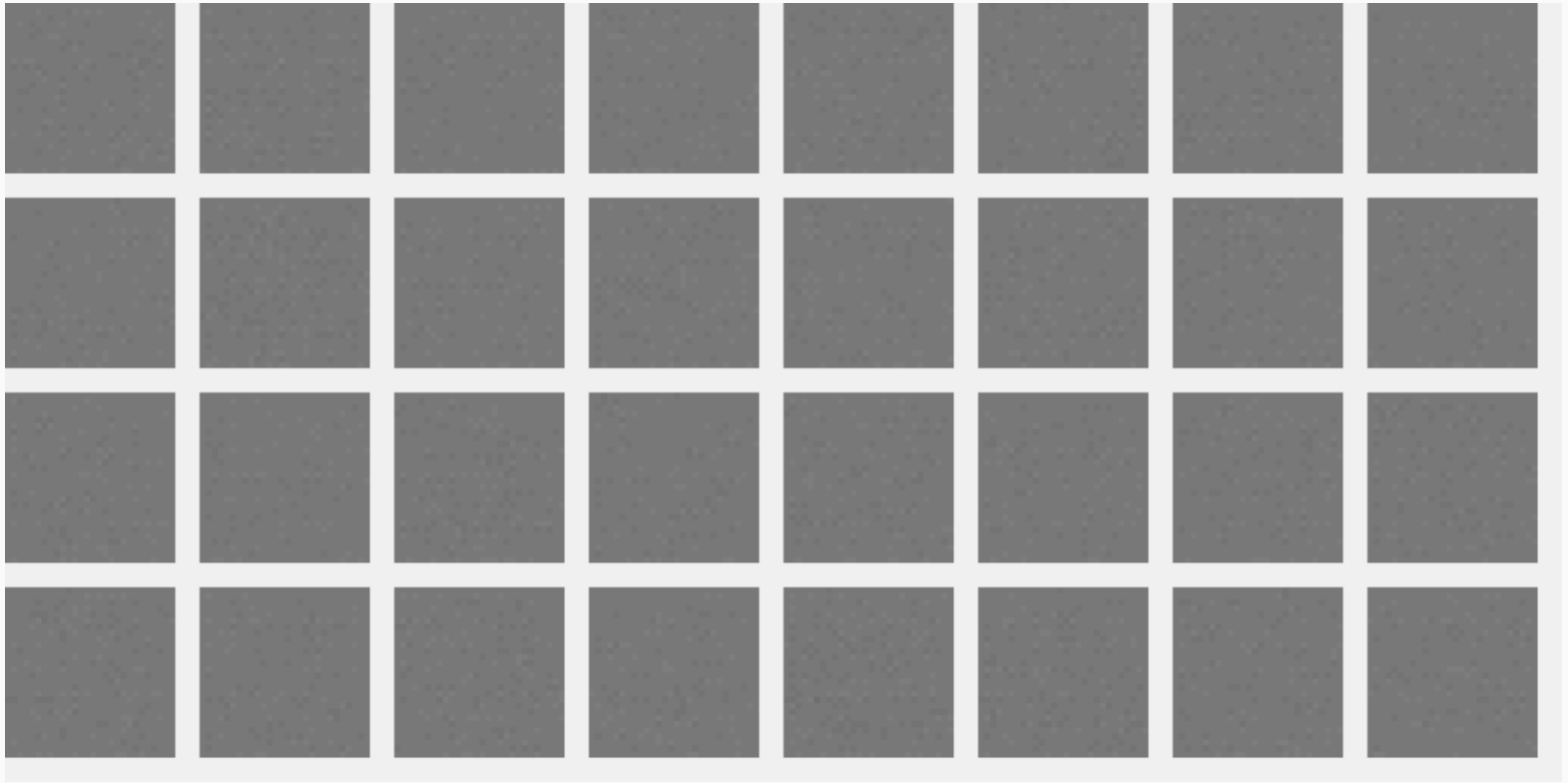


No sigmoid before binary_cross_entropy_with_logits.

> No. This discriminator produces one source-classification logit for each whole image.

▶ Notebook

The same z before and after early learning



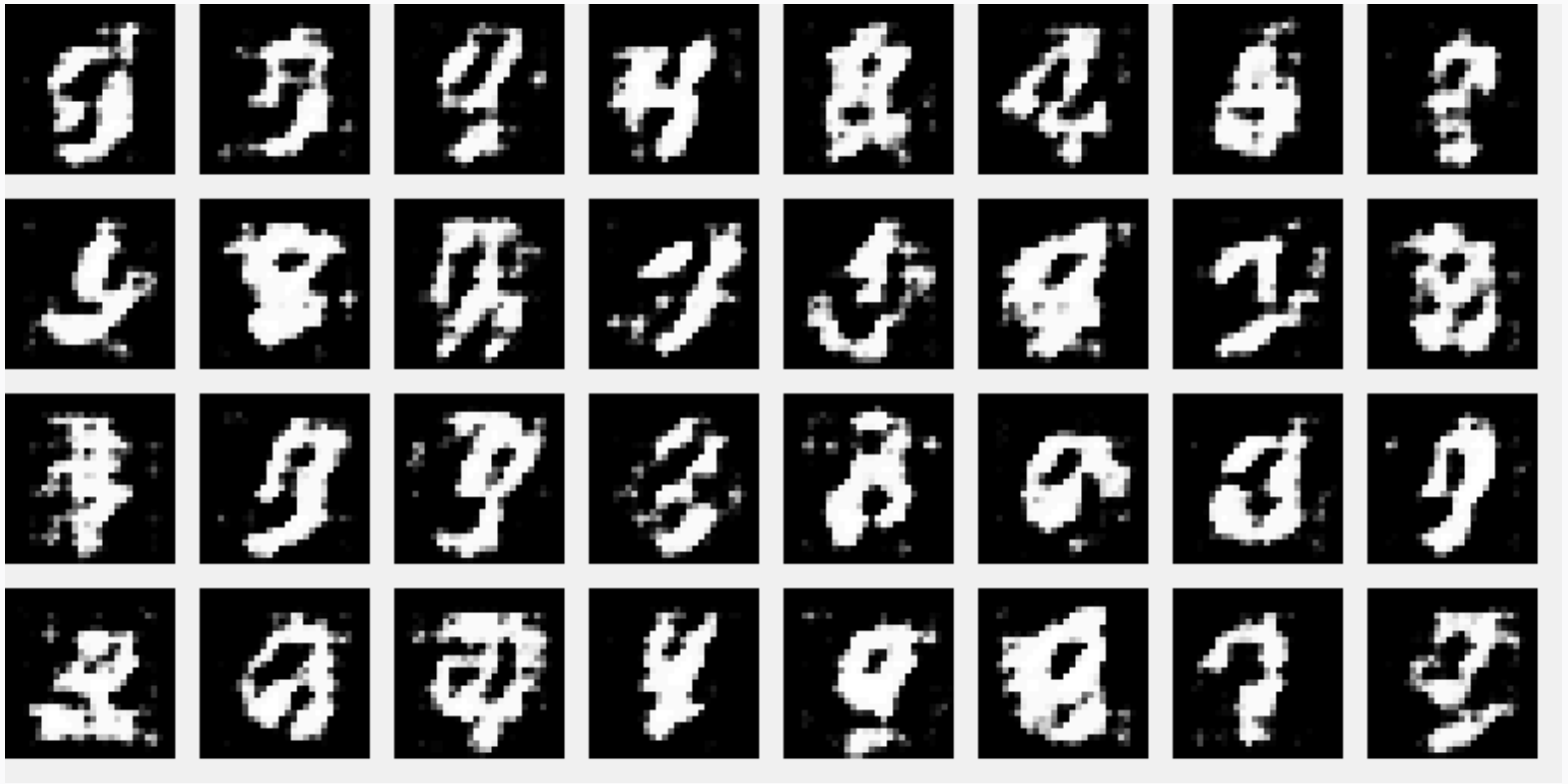
Saved fixed-noise samples · epoch 0

> What is changing between these grids?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

The same z before and after early learning



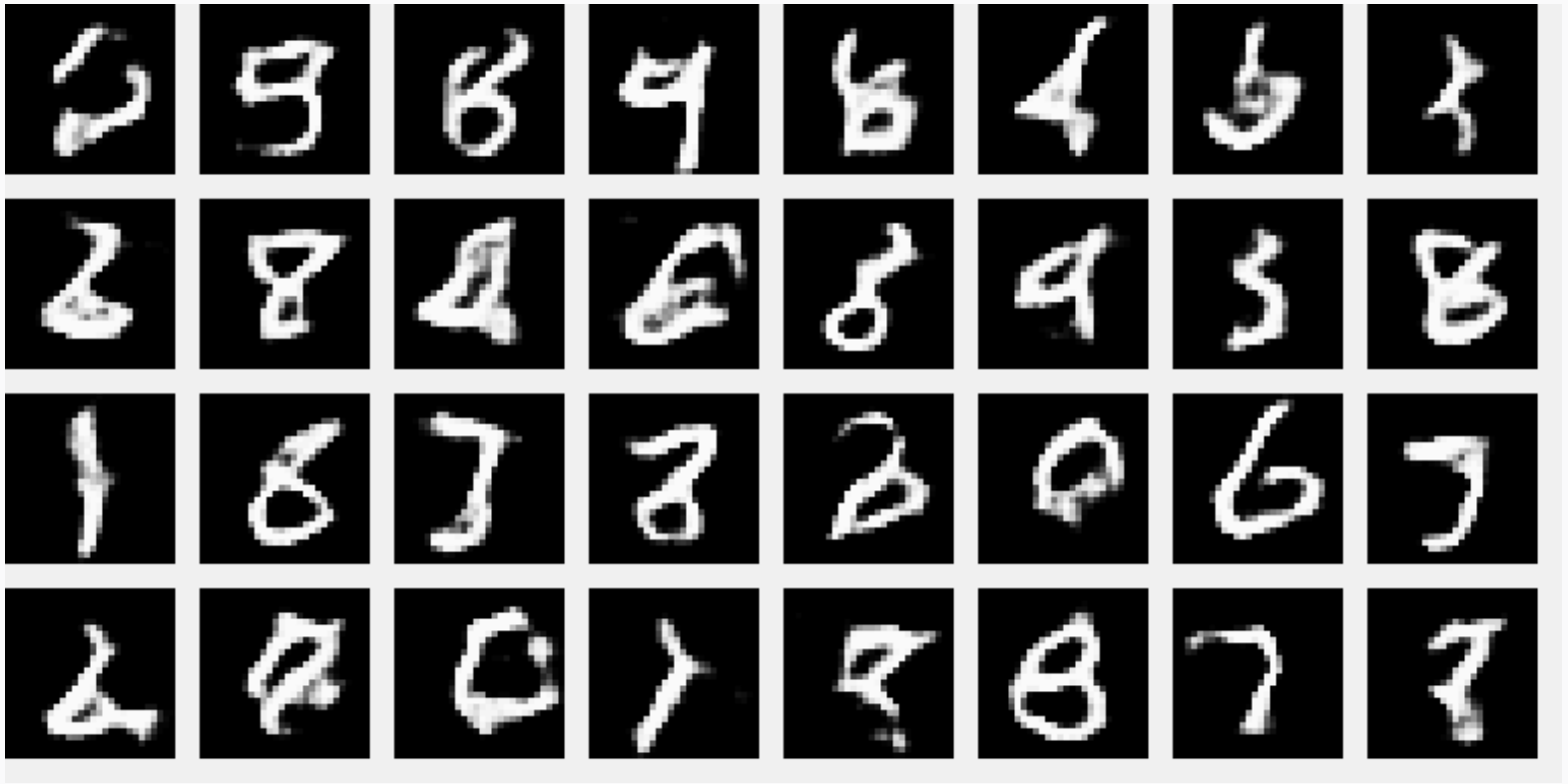
Saved fixed-noise samples · epoch 1

> Generator weights; the latent vectors are held fixed.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Structure emerges, but quality varies

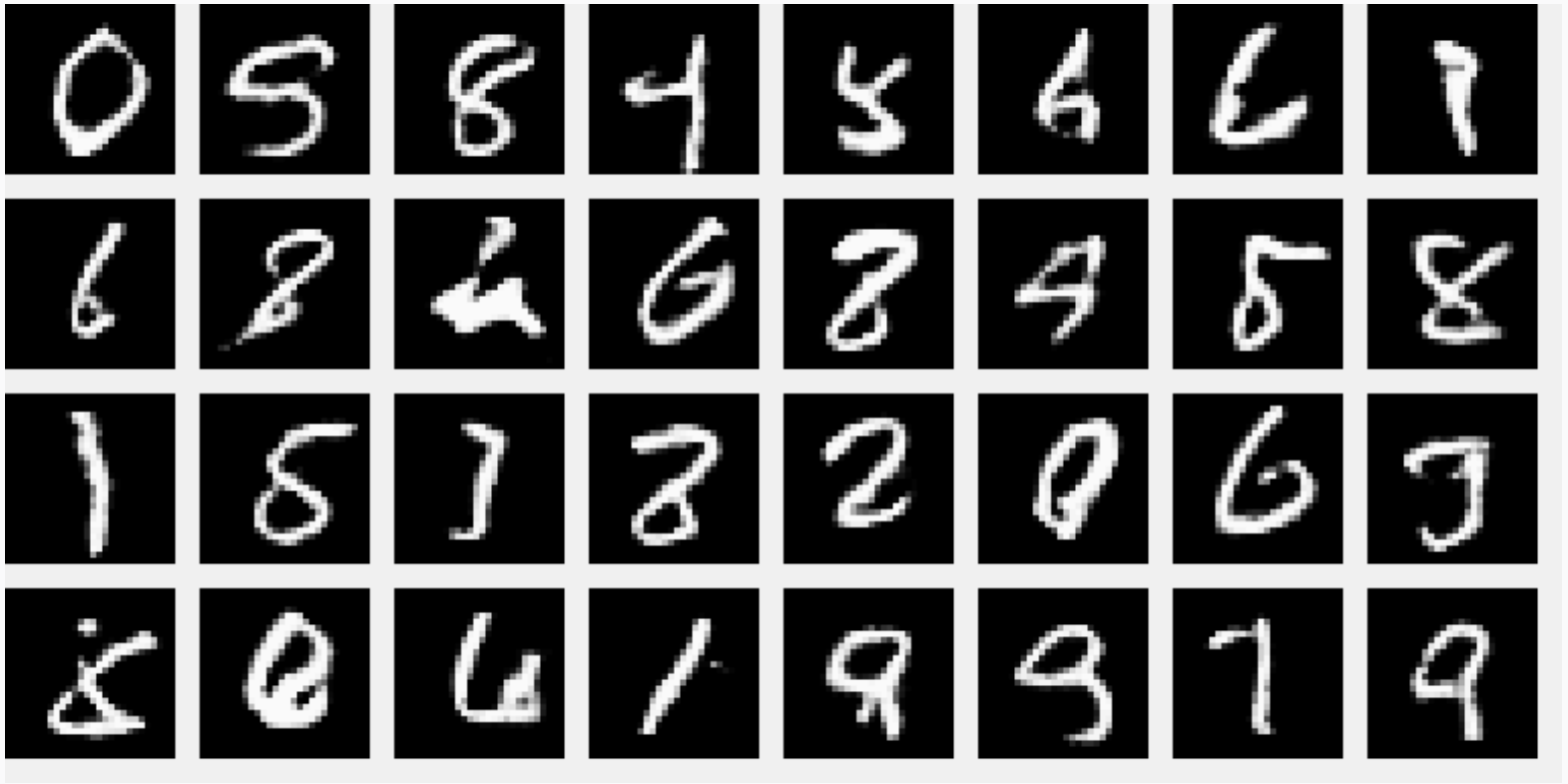


Saved fixed-noise samples · epoch 5

> Which limitations remain visible?

▶ Notebook

Structure emerges, but quality varies



Saved fixed-noise samples · epoch 30 · no quality metric claimed

> Some samples are ambiguous; a fixed grid alone does not establish distributional fidelity.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

What did we reproduce today?

evidence	scope
scalar D and G steps	fresh execution + gradient checks
MNIST samples	replayed saved 30-epoch model
maximum replay difference	0.00000528
full MNIST retraining	documented optional run

> Does successful replay mean we retrained the image GAN?

▶ Notebook

What did we reproduce today?

evidence	scope
scalar D and G steps	fresh execution + gradient checks
MNIST samples	replayed saved 30-epoch model
maximum replay difference	0.00000528
full MNIST retraining	documented optional run

CPU replay can differ slightly from the original device's arithmetic.

> No. It verifies the checkpoint and sample-generation path.

▶ Notebook

Live: inspect who changes

```
checks = checked_neural_step()
print(checks)
# D phase: G change == 0
# G phase: D change == 0
```

> Which two parameter-change checks must be exactly zero?

▶ Notebook

Live: inspect who changes

```
checks = checked_neural_step()
print(checks)
# D phase: G change == 0
# G phase: D change == 0
```

Predict each zero before running the cell.

> G during the D phase and D during the G phase.

▶ Notebook

Exit check: where should detach go?

```
fake = G(z)
score = D(fake)
loss = BCE_logits(score, target)
```

> How does the answer change between the two phases?

▶ Notebook

Exit check: where should detach go?

```
fake = G(z)
score = D(fake)
loss = BCE_logits(score, target)
```

Detach fake for the D loss. Keep the path for the G loss.

> The same forward topology serves different parameter owners and targets.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Two ways to train a sampler

model	training signal	sample after training
VAE	likelihood term + prior KL	z → decoder
GAN	learned discriminator feedback	z → generator

> What did the discriminator replace in the training story?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Two ways to train a sampler

model	training signal	sample after training
VAE	likelihood term + prior KL	z → decoder
GAN	learned discriminator feedback	z → generator

Both require evidence beyond a single appealing sample. Neither loss number is directly comparable to the other.

> It supplies feedback without evaluating a decoder likelihood or an ELBO.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Can we create the training target ourselves?

A GAN learns a critic to judge samples.

> What information becomes available when we create the corruption?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

▶ Notebook

Can we create the training target ourselves?

A GAN learns a critic to judge samples.

Next: add known noise to real images, then learn a prediction task with a known target.

> The clean example, timestep and injected noise are all known during training.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

Optional: the ideal classifier at fixed G

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$

> Under what assumptions is this ideal discriminator statement meaningful?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

Optional: the ideal classifier at fixed G

$$D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_G(x)}$$

$$p_G = p_{\text{data}} \implies D^*(x) = \frac{1}{2}$$

> Unlimited classifier capacity and exact optimisation at fixed distributions; finite neural training need not reach it.

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

Optional: frozen parameters are not all model state

BatchNorm running statistics and dropout behaviour depend on training/evaluation mode.

> Does requires_grad_(False) switch a network to evaluation mode?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

Optional: frozen parameters are not all model state

BatchNorm running statistics and dropout behaviour depend on training/evaluation mode.

Our scalar checks have no mutable buffers. The MNIST D uses dropout and no BatchNorm.

> No. Parameter gradients and train/eval mode are separate controls.

Optional: changes to the game are experiments

Capacity, update ratio, regularisation and architecture can change training behaviour.

> Does one stabilised run prove a remedy always works?

Original teaching construction · Goodfellow et al. (2014), arxiv.org/abs/1406.2661

Optional: changes to the game are experiments

Capacity, update ratio, regularisation and architecture can change training behaviour.

Change one factor, retain seeds and diagnostics, and report what the run actually shows.

> No. It supports the measured configuration only.