U1.T1 — Change of Variables and Coupling Flows

Flow-Based Generative Models · UFRJ · 2026.2

01 The exact-likelihood contract

Where this session sits

Unit	Strategy	What it costs
U1 — today	invertible by construction	what the map is allowed to be
U2	volume change by calculus	a solver inside every training step
U3	refuse to play	nothing yet

Today is the first row. It is also the first session where a model gets trained.

What UP.T1 left on the table

A generative model is a pushforward of noise, and training is maximum likelihood:

$$p_\theta = (\psi^\theta)_\# p_0, \qquad \arg\max_\theta \ \mathbb{E}_{x_1 \sim q}[\log p_\theta(x_1)] = \arg\min_\theta \ \mathrm{KL}(q \,\|\, p_\theta).$$

Therefore training needs $\log p_\theta(x_1)$ **at data points.** The definition only gives you samples.

The problem, stated honestly

For a generic network ψ^θ, the density at x_1 is an integral over every noise point that lands there:

$$p_\theta(x_1) = \int p_0(x_0)\,\delta\left(x_1 - \psi^\theta(x_0)\right)\,\mathrm{d}x_0.$$

No closed form. Preimages may be empty, or infinite, or the output may lie on a surface with no density at all.

Two escape routes: **bound it** (the variational autoencoder; one line, not today), or **make it exact by construction**. This unit takes the second.

The session question

What must ψ^θ satisfy for $\log p_\theta(x_1)$ to be exactly computable?

This stays on the board until the end. A second question joins it later, and that one is not answered today.

02 The change-of-variables theorem

In one dimension, you already proved it

RESULT (CHANGE OF VARIABLES, 1D — UP.A §4.3)

$y = g(x)$ with g differentiable and strictly monotone:

$$p_y(y) = p_x(g^{-1}(y)) \left| \frac{\mathrm{d}\, g^{-1}}{\mathrm{d}y}(y) \right| .$$

The absolute value is the content: **mass is conserved, density is not.** Where g stretches, the density falls by the same factor.

PS1.1.8 made you compute it for tanh. The primer promised the d-dimensional version as PS1.3. Here it is.

The theorem in d dimensions

RESULT (THE CHANGE-OF-VARIABLES THEOREM)

ψ a diffeomorphism of $\mathbb{R}^d$, $x \sim p$, $y = \psi(x)$. Then

$$(\psi_{\#}p)(y) = p(\psi^{-1}(y)) \left|\det J_{\psi^{-1}}(y)\right| = \frac{p(x)}{\left|\det J_{\psi}(x)\right|}, \qquad x = \psi^{-1}(y).$$

Hypotheses, displayed: ψ is a **bijection**, ψ is **differentiable**, ψ^{-1} is **differentiable**.

The two forms agree because $J_{\psi^{-1}}(y) = J_{\psi}(x)^{-1}$. Statement as in (Papamakarios et al. 2021, sec. 2.1).

Which form this course writes

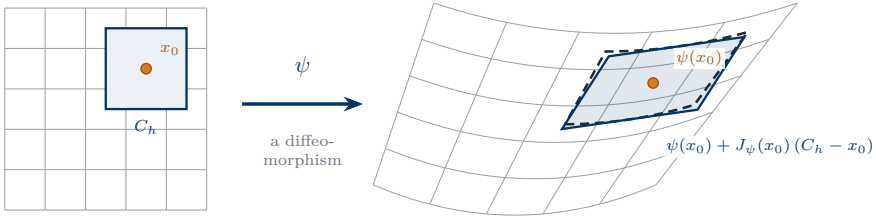
$$\log p_\theta(x_1) \;=\; \log p_0(x_0) \;-\; \log\left|\det J_{\psi^\theta}(x_0)\right|, \qquad x_0 \;=\; (\psi^\theta)^{-1}(x_1).$$

Log form, forward Jacobian, and ψ^θ **generates** (noise to data), as in UP.T1.

Read it as a recipe: **undo the map** to find x_0, evaluate $\log p_0$ there, **subtract the log-volume factor**.

The papers write the map the other way (f: data to latent, $= (\psi^\theta)^{-1}$). Translate before you read.

$|\det J_\psi|$ is local volume distortion



source: $x \sim p$

a cube of side h around x_0: volume h^d, mass $\approx p(x_0)\,h^d$

image: $y = \psi(x) \sim \psi_{\#}\,p$

dark dashed: the true image $\psi(C_h)$, curved. Blue solid: its linearization, a parallelepiped of volume $|\det J_\psi(x_0)|\,h^d$

Same mass in both cells: $(\psi_{\#}p)(\psi(x_0)) \cdot |\det J_\psi(x_0)|\,h^d \;\approx\; p(x_0) \cdot h^d$

so the density at $\psi(x_0)$ is the density at x_0 divided by the volume factor $|\det J_\psi(x_0)|$.

Same mass, different volume

$$\underbrace{(\psi_{\#}p)(\psi(x_o))}_{\text{density at image}} \cdot \underbrace{|\det J_\psi(x_o)|\, h^d}_{\text{volume of image}} \approx \underbrace{p(x_o)}_{\text{density at source}} \cdot \underbrace{h^d}_{\text{volume of source}}$$

Every noise point in the cube lands in the image cell, and no other does, because ψ is a bijection.

The determinant is the exchange rate between volume and density. Where the map expands, the density drops by the same factor.

Proof sketch, with the gaps named

1. **Linear maps move volume by** $|\det A|$**:** $\mathrm{vol}(AB) = |\det A|\,\mathrm{vol}(B)$. Gap 1: prove it — factor A into elementary matrices.
2. **A diffeomorphism is locally linear:** $\psi(x) = \psi(x_0) + J_\psi(x_0)(x - x_0) + r(x)$, $r(x) = o(|x - x_0|)$, so $\mathrm{vol}(\psi(C_h)) = |\det J_\psi(x_0)|\, h^d\,(1 + o(1))$. Gap 2: the $o(1)$ must be uniform — continuity of the Jacobian.
3. **Mass conservation:** $\mathbb{P}[y \in \psi(C_h)] = \mathbb{P}[x \in C_h]$; divide by h^d, let $h \to 0$. Gap 3: this assumes y has a continuous density. The honest proof shows the substitution rule first, then reads the density off it — dominated convergence.

Waved hands, **named as waved hands**. Closing them is PS1.3.

PS1.3 — your first proof-completion

PS1.3 — PROVE THE THEOREM, BY THE LINEARIZATION ROUTE

(a) Linear. $\mathrm{vol}(AB) = |\det A|\,\mathrm{vol}(B)$ for boxes B, via elementary matrices.
(b) Local. $\mathrm{vol}(\psi(C_h))/h^d \to |\det J_\psi(x_0)|$: sandwich $\psi(C_h)$ between two scaled copies of the parallelepiped $J_\psi(x_0)(C_h - x_0)$.
(c) Global. Cut a box into cubes, apply (b), pass to the limit with dominated convergence: $\int_{\psi(B)} g = \int_B (g \circ \psi)\,|\det J_\psi|$.
(d) Read off the density, both forms, and the log form.

Route fixed on purpose: linearization, not measure-theoretic pushforward machinery. The notes carry the hints.

The fine print

The theorem asks three things of ψ:

1. ψ is a **bijection** of $\mathbb{R}^d$ onto $\mathbb{R}^d$;
2. ψ is **differentiable**;
3. ψ^{-1} is **differentiable**.

Together: a **diffeomorphism**. Everything built today inherits these hypotheses.

03 Compositions and the two costs

Compose invertible maps

$$\psi^\theta = \psi_K \circ \cdots \circ \psi_1, \qquad (\psi^\theta)^{-1} = \psi_1^{-1} \circ \cdots \circ \psi_K^{-1}, \qquad \det J_{\psi^\theta}(x_0) = \prod_k \det J_{\psi_k}(z_{k-1})$$

Diffeomorphisms compose. The inverse undoes the layers in reverse; the chain rule multiplies the determinants.

Name the intermediate points once:

$$z_0 = x_0, \qquad z_k = \psi_k(z_{k-1}), \qquad z_K = x_1.$$

A sample **flows** through the layers. The inverse **normalizes** data back to the source.

The log-likelihood telescopes

$$\log p_\theta(x_1) \;=\; \log p_0(z_0) \;-\; \sum_{k=1}^{K} \log \left| \det J_{\psi_k}(z_{k-1}) \right|, \qquad z_K = x_1, \;\; z_{k-1} = \psi_k^{-1}(z_k).$$

One board line. The chain is computed by **running the inverse**, from the data point down to the noise point.

Every term must be cheap: for every layer, at every data point, at every training step.

The two costs

THE TWO COSTS (V1)

Operation	Direction	Needs, per layer	Cost driver
Sampling	forward, $x_0 \to x_1$	evaluate ψ_k	one pass per layer
Density / training	inverse, $x_1 \to x_0$	ψ_k^{-1} **and** $\log\lvert\det J_{\psi_k}\rvert$	the inverse, and the log-det

Versioned: U1.T2 adds a row per architecture; U2.T2 changes what the cells contain; U3.L1 touches neither row.

Why a naive layer is dead on arrival

A dense $d \times d$ Jacobian: $O(d^2)$ to store, $O(d^3)$ to take the determinant of — per layer, per data point, per step.

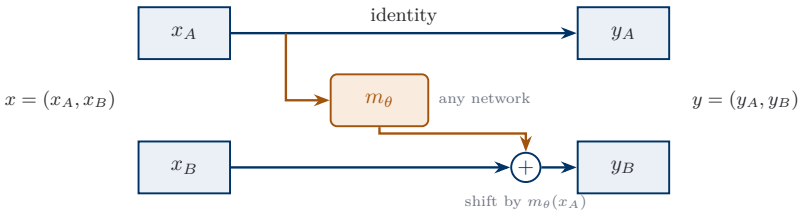
32×32 colour image: $d = 3072$, so $d^3 \approx 3 \times 10^{10}$.

And the inverse of a generic network is not available at all.

The whole architectural zoo of this unit is a list of tricks to make both rows cheap simultaneously.

04 NICE: additive coupling

A coupling layer, as a graph



One layer: input x, output y. In the chain of block C, $x = z_{k-1}$ and $y = z_k$.

Split the coordinates

Partition $x = (x_A, x_B)$, $d_A + d_B = d$. Take **any** network $m_\theta : \mathbb{R}^{d_A} \to \mathbb{R}^{d_B}$:

DEFINITION (ADDITIVE COUPLING LAYER — NICE)

$$y_A = x_A, \qquad y_B = x_B + m_\theta(x_A).$$

Half passes through untouched. The other half is shifted by an amount that depends on the untouched half only (Dinh, Krueger, and Bengio 2015, sec. 3.2).

The Jacobian, on the board

Order the coordinates (x_A, x_B) and differentiate block by block:

$$J_\psi(x) = \begin{bmatrix} I_{d_A} & 0 \\ \dfrac{\partial m_\theta}{\partial x_A}(x_A) & I_{d_B} \end{bmatrix} \quad \Longrightarrow \quad \det J_\psi(x) = 1.$$

Block lower-triangular: the determinant is the product of the diagonal blocks. The expensive block **never enters**.

Inverse, free: $x_A = y_A, \quad x_B = y_B - m_\theta(y_A)$.

The trick the whole family rides on

Invertibility of the whole does not require invertibility of the parts.

To undo the layer you evaluate m_θ at y_A, which you have, because $y_A = x_A$. You never invert m_θ.

So m_θ can be as deep, as wide, and as non-invertible as you like. Every coupling architecture in this unit rides on this.

The honest limitation

$\det J \equiv 1$ means the layer **preserves volume**. So does any stack of them.

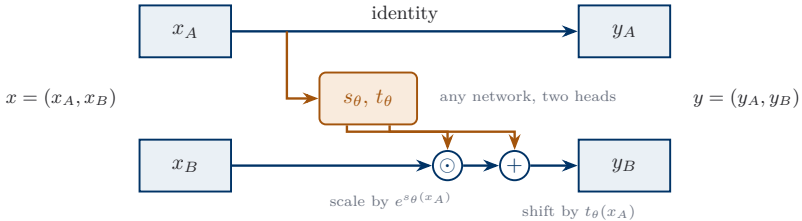
A volume-preserving flow cannot concentrate or spread mass. A real dataset has a different spread from $\mathcal{N}(0, I_d)$.

NICE's fix: one final diagonal **scaling layer**, $y_i = S_{ii}\, x_i$, with $\log|\det J_\psi(x)| = \sum_i \log|S_{ii}|$. A global rescaling, once.

The next block makes the rescaling **local**. That is the whole difference.

05 RealNVP: affine coupling

The same graph, with a scale



$$y_A = x_A, \qquad y_B = x_B \odot e^{s_\theta(x_A)} + t_\theta(x_A)$$

to undo the layer, evaluate s_θ, t_θ at y_A (which equals x_A), subtract and divide: the network is never inverted.

Same untouched half, same conditioner reading it. One new operation on the path of x_B: a scale before the shift.

Edit one block

Keep the split and the pass-through half. Replace the shift by a shift **and a scale**, both read off x_A:

DEFINITION (AFFINE COUPLING LAYER — REALNVP)

$$y_A = x_A, \qquad y_B = x_B \odot \exp\big(s_\theta(x_A)\big) + t_\theta(x_A).$$

$s_\theta, t_\theta \colon \mathbb{R}^{d_A} \to \mathbb{R}^{d_B}$ arbitrary networks; $\odot$ elementwise. The exponential keeps every scale positive (Dinh, Sohl-Dickstein, and Bengio 2017, sec. 3.2).

The Jacobian, edited in place

NICE: $y_B = x_B + m_\theta(x_A)$

$$J_\psi(x) \;=\; \begin{bmatrix} I_{d_A} & 0 \\ \dfrac{\partial m_\theta}{\partial x_A} & I_{d_B} \end{bmatrix}$$

$$\det J_\psi(x) \;=\; 1 \cdot 1 \;=\; 1$$

block-triangular, so the determinant is the product of the diagonal blocks. The gray block is never needed.

edit one block

RealNVP: $y_B = x_B \odot e^{s_\theta(x_A)} + t_\theta(x_A)$

$$J_\psi(x) \;=\; \begin{bmatrix} I_{d_A} & 0 \\ \dfrac{\partial y_B}{\partial x_A} & \operatorname{diag}\left(e^{s_\theta(x_A)}\right) \end{bmatrix}$$

$$\log|\det J_\psi(x)| \;=\; \textstyle\sum_{j=1}^{d_B} s_\theta(x_A)_j$$

the edited block is diagonal, so its determinant is the product of its entries: a sum read off one forward pass of s_θ.

Read off a forward pass

$$\log \left| \det J_{\psi}(x) \right| = \sum_{j=1}^{d_B} s_\theta(x_A)_j$$

A sum read off one forward pass of s_θ. No $d \times d$ matrix formed, no determinant computed.

Inverse, closed form: $x_A = y_A, \quad x_B = (y_B - t_\theta(y_A)) \odot \exp(-s_\theta(y_A))$.

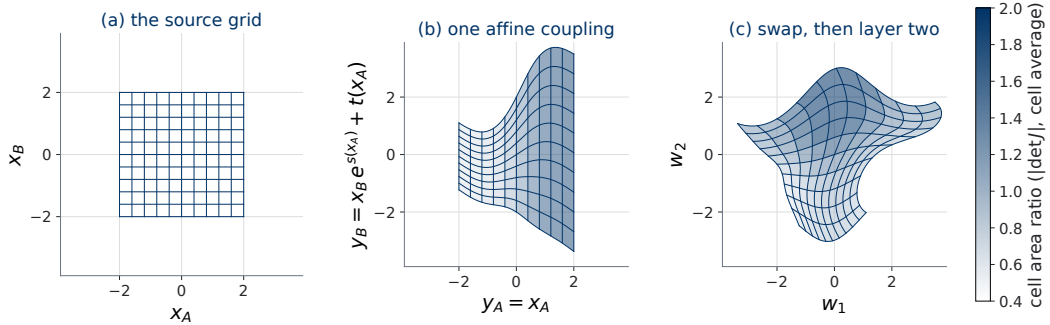
The sum depends on $x_A = y_A$ only, so it is available on **both** sides of the layer — the inverse pass, which training runs, reads it for free.

NICE and RealNVP, side by side

	NICE (additive)	RealNVP (affine)
Layer	$y_B = x_B + m_\theta(x_A)$	$y_B = x_B \odot e^{s_\theta(x_A)} + t_\theta(x_A)$
$\log\lvert\det J_\psi\rvert$	0	$\sum_j s_\theta(x_A)_j$
Inverse	subtract; one pass of m_θ	subtract, divide; one pass of s_θ, t_θ
Volume	preserved; a global scaling layer at the end	changed locally, per layer
Untouched half	x_A, read by the conditioner	x_A, read by the conditioner

Both rows of the two-costs table are one conditioner pass per layer, in both directions, for both layers.

What one layer does to a grid



Shading is the cell's area ratio, $|\det J|$ averaged over the cell: constant down each column, because it depends on x_A alone.

A numerical aside, for the lab

e^s overflows in single precision near $s \approx 88$. e^{-s} in the inverse divides by almost zero when s is large.

Early in training s_θ is a random network, and both happen.

Implementations **bound the scale**: $s = c \cdot \tanh(\tilde{s})$ before the exponential.

One slide. The lab (U1.L1) shows the loss curve when you forget.

The training loss, in full

$$\mathcal{L}(\theta) = -\,\mathbb{E}_{x_1 \sim q}\Big[\log p_0(z_0) - \sum_{k=1}^{K} \log\big|\det J_{\psi_k}(z_{k-1})\big|\Big],$$

$$z_K = x_1, \qquad z_{k-1} = \psi_k^{-1}(z_k).$$

The subscript on $\mathbb{E}$ is mandatory, and this is the first loss of the course where it is enforced. Each log-det is a sum of scales at the untouched half of z_{k-1}.

PS1.4 is exactly the bookkeeping of this expression for a three-layer RealNVP. Assigned now.

PS1.4 — the bookkeeping

PS1.4 — THREE AFFINE COUPLING LAYERS, D = 4

$\psi^\theta = \psi_3 \circ \psi_2 \circ \psi_1$; coordinates of a vector written $(\xi_1, \ldots, \xi_4)$ (x_1 stays the data point). Layers 1 and 3 condition on (ξ_1, ξ_2) and move (ξ_3, ξ_4); layer 2 the reverse.

(a) Write the inverse pass: z_2, z_1, z_0 from a data point, in the order computed.

(b) Write $\log p_\theta(x_1)$ fully explicitly: every log-det as its sum of scales, every scale at the correct intermediate point.

(c) Write the minibatch loss, sample mean in place of the expectation, subscript kept.

(d) Replace ψ_2 by a swap of the halves. Redo (b). Which term changes?

Mechanical on purpose: the notation of the loss through your hands once, before U1.L1 puts it through code.

06 Masks, partitions, permutations

x_A is never touched, so alternate

A coupling layer leaves x_A exactly as it found it. Same split forever means a model of x_B only.

- **Alternate the split**: layer k moves B, layer $k+1$ moves A. Two layers move everything; three let everything influence everything.
- **Permute between layers**: a fixed P between couplings. Bijection, inverse $P^\top$, $|\det P| = 1$, so its log-det is **zero**.

Check the permutation in your head now; check it in code next session (U1.L1).

For images: binary **masks**, checkerboard and channel-wise. One line; that is U1.L2.

Depth as the lever

With alternation in place, a stack of K affine coupling layers moves every coordinate, and the composed Jacobian is dense once $K \geq 3$.

The natural next belief: with enough layers, a coupling flow reaches anything. **Asserted today. Not proven.**

How expressive is a stack of these, really?

Written on the board next to the session question. Answered in U1.T2.

07 Training loop and close

The training loop, in full

MLE FOR A FLOW — THE WHOLE LOOP

```
for x1 in loader:                          # minibatch, x1 ~ q
    loss = -flow_log_prob(x1).mean()  # minus mean log p
    opt.zero_grad()
    loss.backward()
    opt.step(); sched.step()
```

`flow_log_prob` is the telescoped log-likelihood: inverse chain, plus $\log p_0(z_0)$, minus the K sums of scales. You write it in U1.L1. `opt`: AdamW, LR 3×10^{-4}, weight decay 0.01; `sched`: warmup, then cosine.

Bootcamp readers: `training-loop-template` with a new loss line, under `optimizer-schedule-defaults`. For everyone else, the box is the whole definition.

Reading the board

On the board	Status
The session question — what must ψ^θ satisfy?	**Answered:** a diffeomorphism, built from layers whose inverse and log-det are cheap.
The depth question — how expressive is a stack?	**Open.** U1.T2.
The fine print — bijection, differentiable, both ways	**Unexamined.** U1.T2.

What comes next

- **Next session (U1.L1):** build it. An affine coupling layer and a permutation from scratch, the sum of scales checked against an autodiff Jacobian, trained on 2D toy data by the loop you just saw.
- **The theory session after (U1.T2):** the rest of the zoo, and where its expressivity claims break.

Assigned today: **PS1.3** (the proof) and **PS1.4** (the bookkeeping).

References

Dinh, Laurent, David Krueger, and Yoshua Bengio. 2015. “NICE: Non-Linear Independent Components Estimation.” In *International Conference on Learning Representations, Workshop Track*. https://arxiv.org/abs/1410.8516.

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio. 2017. “Density Estimation Using Real NVP.” In *International Conference on Learning Representations*. https://arxiv.org/abs/1605.08803.

Papamakarios, George, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2021. “Normalizing Flows for Probabilistic Modeling and Inference.” *Journal of Machine Learning Research* 22 (57): 1–64. https://arxiv.org/abs/1912.02762.