# The MadNIS Reloaded

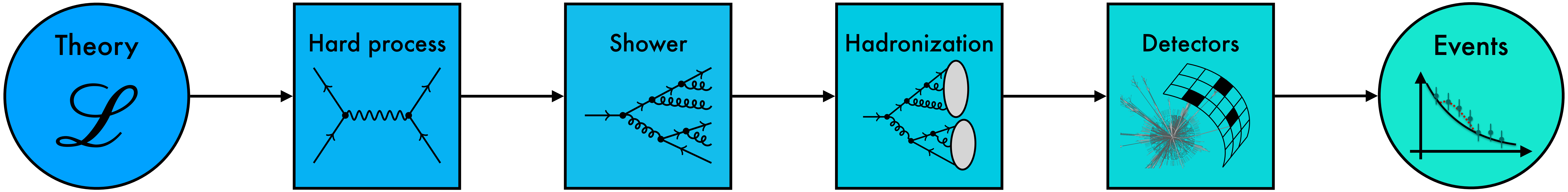
**Theo Heimel**
December 2023

Institut für theoretische Physik
Universität Heidelberg

[2311.01548] TH, Huetsch, Maltoni, Mattelaer, Plehn, Winterhalder

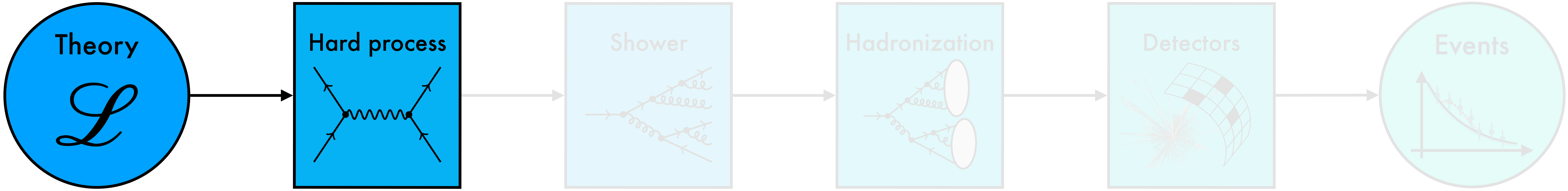
# Introduction

**How can we prevent MC event generation from becoming a bottleneck in future LHC runs?**



# Introduction

**How can we prevent MC event generation from becoming a bottleneck in future LHC runs?**




Differential cross section known from QFT:

$$d\sigma \sim \text{pdf}(x) \cdot |\mathscr{M}(x)|^2 \cdot d\Phi$$

Total cross section:

$$\sigma = \int_\Phi d\sigma$$

Monte Carlo integration and sampling from differential cross section

↓

accelerate with deep generative models

Exact sampling ensured by known likelihood

↓

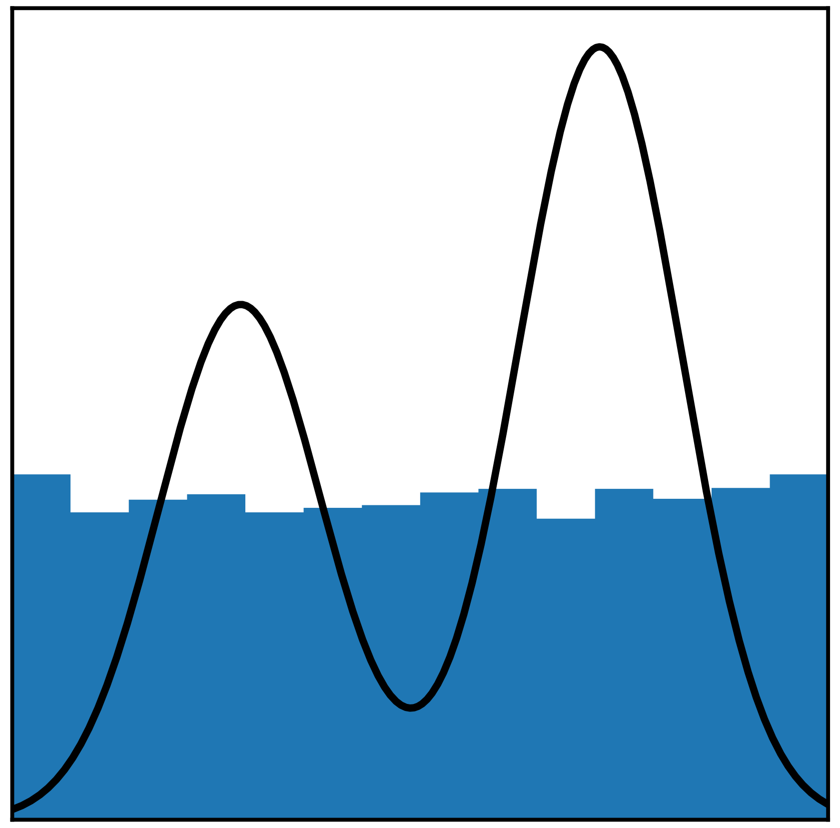
better model
=
faster sampling

# Monte Carlo Integration

$$I = \int \mathrm{d}x \, f(x)$$

# Monte Carlo Integration

$$I = \int \mathrm{d}x\, f(x)$$

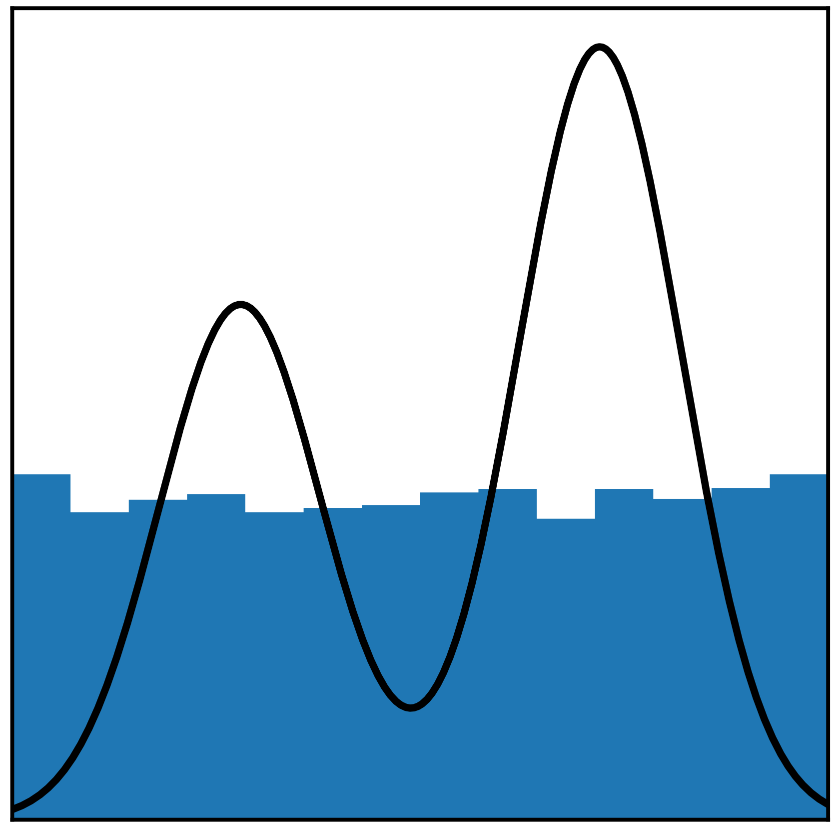


**Flat sampling**
inefficient

$$I = \langle f(x) \rangle_{x \sim p(x)}$$

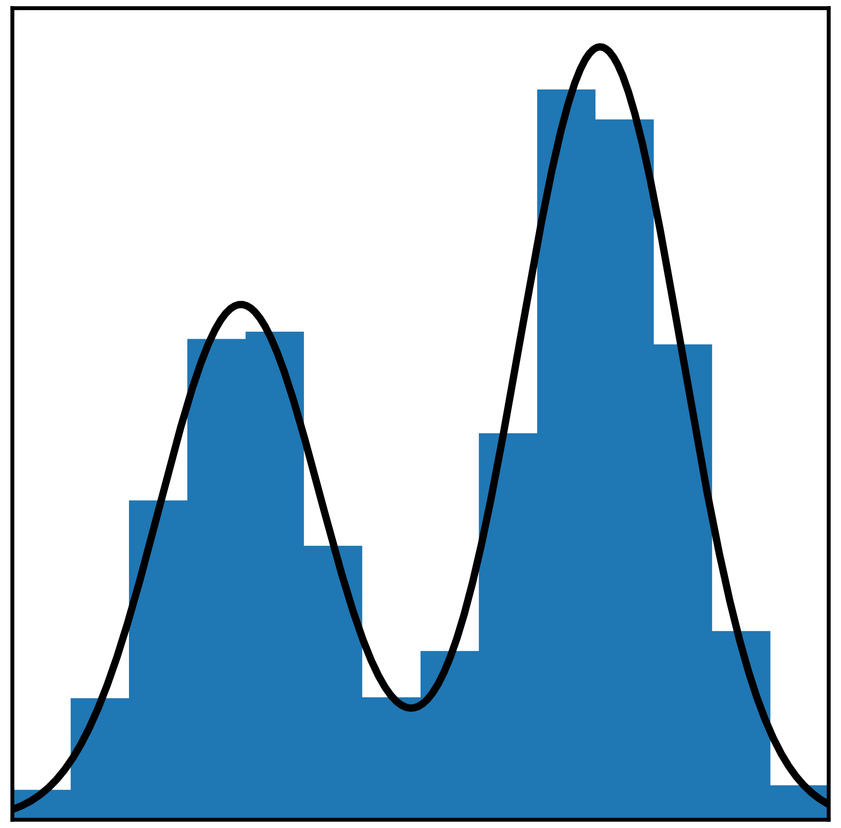
# Monte Carlo Integration

$$I = \int \mathrm{d}x\, f(x)$$



**Flat sampling**
inefficient

$$I = \langle f(x) \rangle_{x\sim p(x)}$$

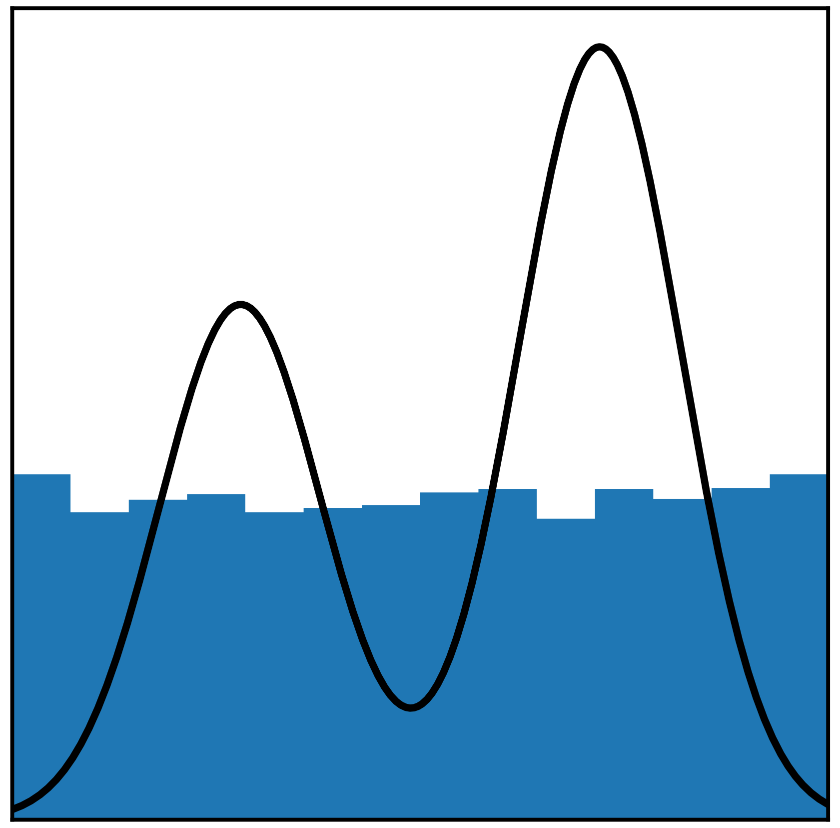


**Importance sampling**
Find mapping close to integrand

$$I = \left\langle \frac{f(x)}{g(x)} \right\rangle_{x\sim g(x)}$$

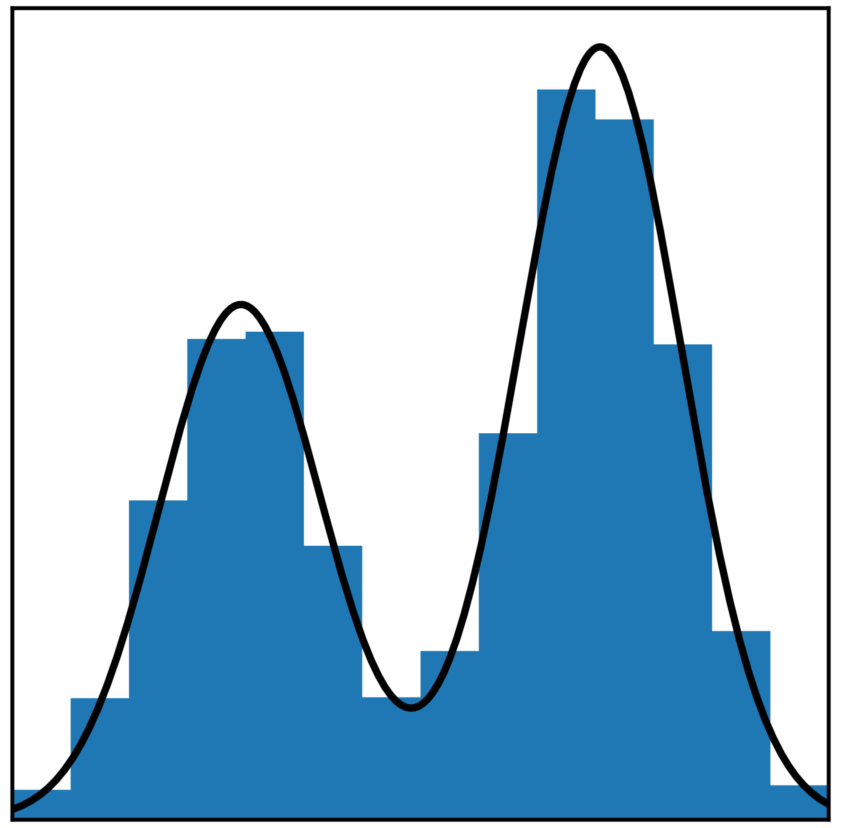
# Monte Carlo Integration

$$I = \int \mathrm{d}x\, f(x)$$



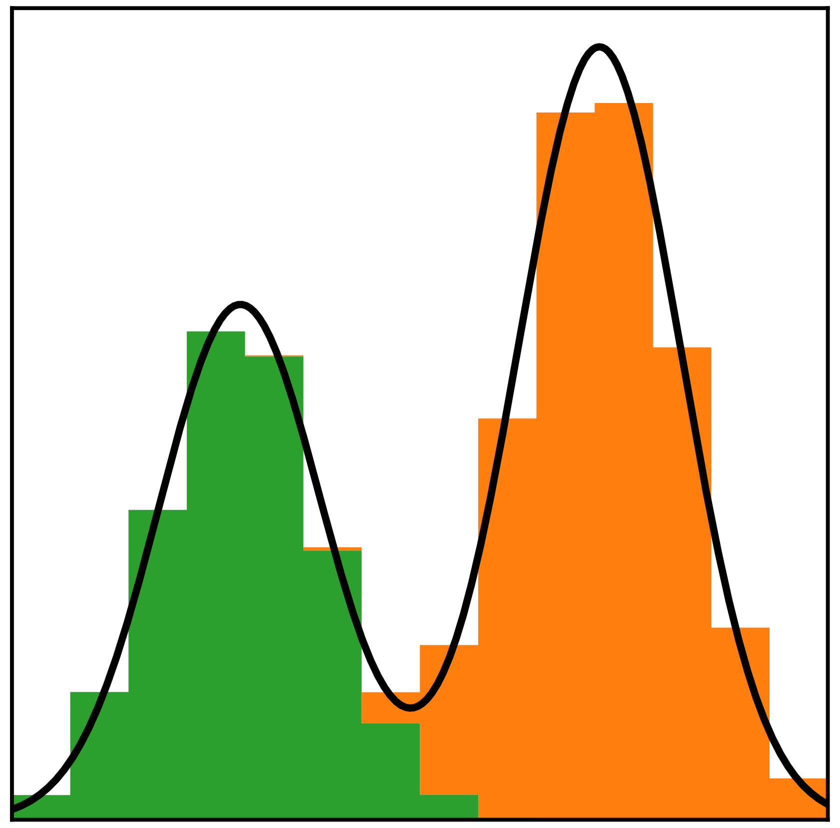
**Flat sampling**
inefficient

$$I = \langle f(x) \rangle_{x \sim p(x)}$$



**Importance sampling**
Find mapping close to integrand

$$I = \left\langle \frac{f(x)}{g(x)} \right\rangle_{x \sim g(x)}$$



**Multi-channeling**
one mapping for each channel

$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

# PS Integration in Madgraph

**How can we make event generation faster?**

Efficient integration and sampling from differential cross section

$$\mathrm{d}\sigma = \frac{1}{\text{flux}}\, \mathrm{d}x_a\, \mathrm{d}x_b\, f(x_a) f(x_b)\, \mathrm{d}\Phi_n \left\langle\, |M_{\lambda,c,\ldots}(p_a, p_b \mid p_1, \ldots, p_n)|^2 \,\right\rangle$$

**Sum over channels**

MadGraph: build channels from Feynman diagrams

**Integrand**

MadGraph: $\mathrm{d}\sigma/\mathrm{d}x$

$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

**Channel weights**

MadGraph: $\alpha_i \sim |M_i|^2$

or $\alpha_i \sim \prod |p_k^2 - m_k^2 - \mathrm{i} M_k \Gamma_k|^{-2}$

**Channel mappings**

MadGraph: use propagators, …

Refine with VEGAS

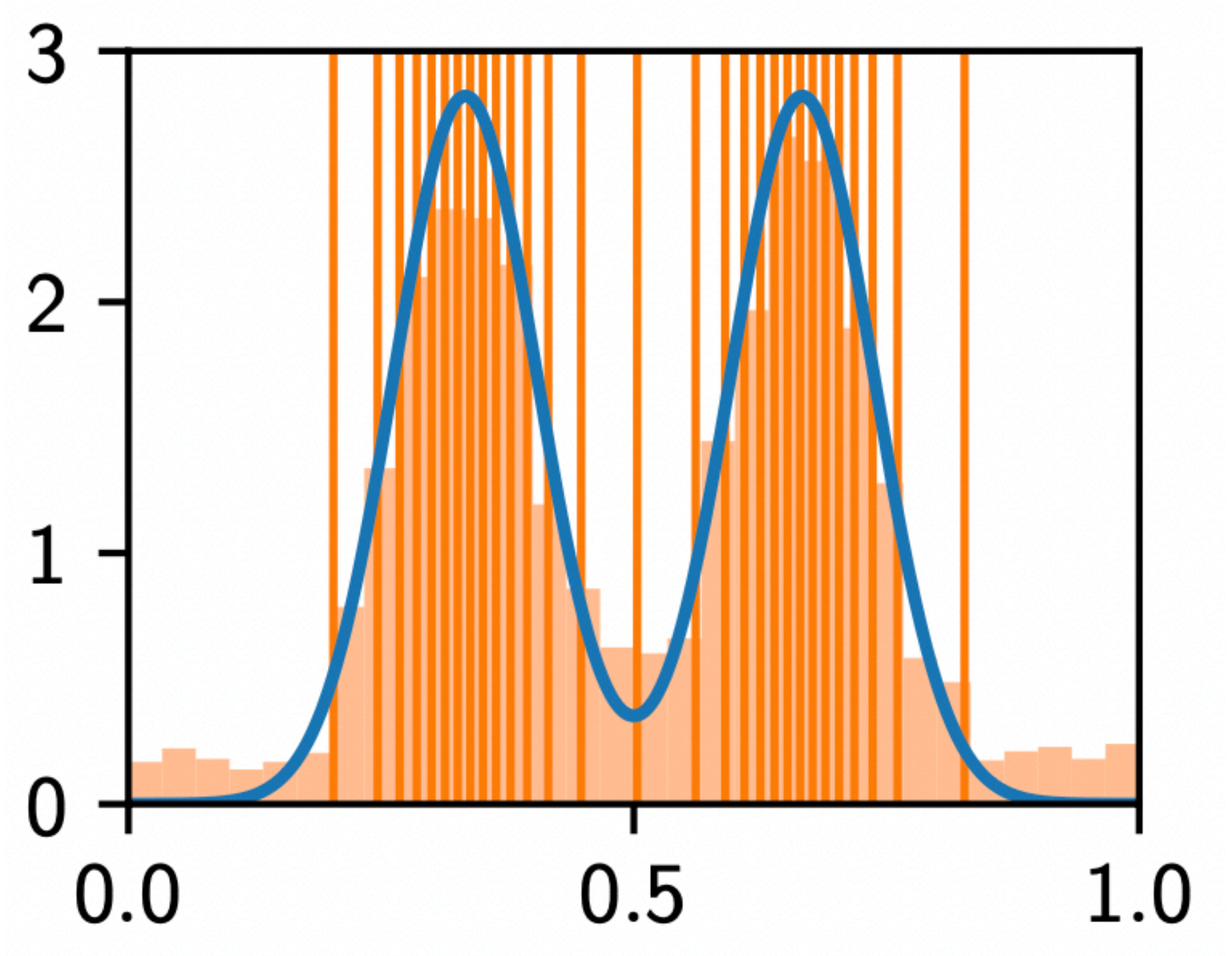
(factorized, histogram based importance sampling)

# VEGAS algorithm

Factorize probability

$$p(x) = p(x_1) \cdots p(x_n)$$

Fit bins with equal probability and varying width



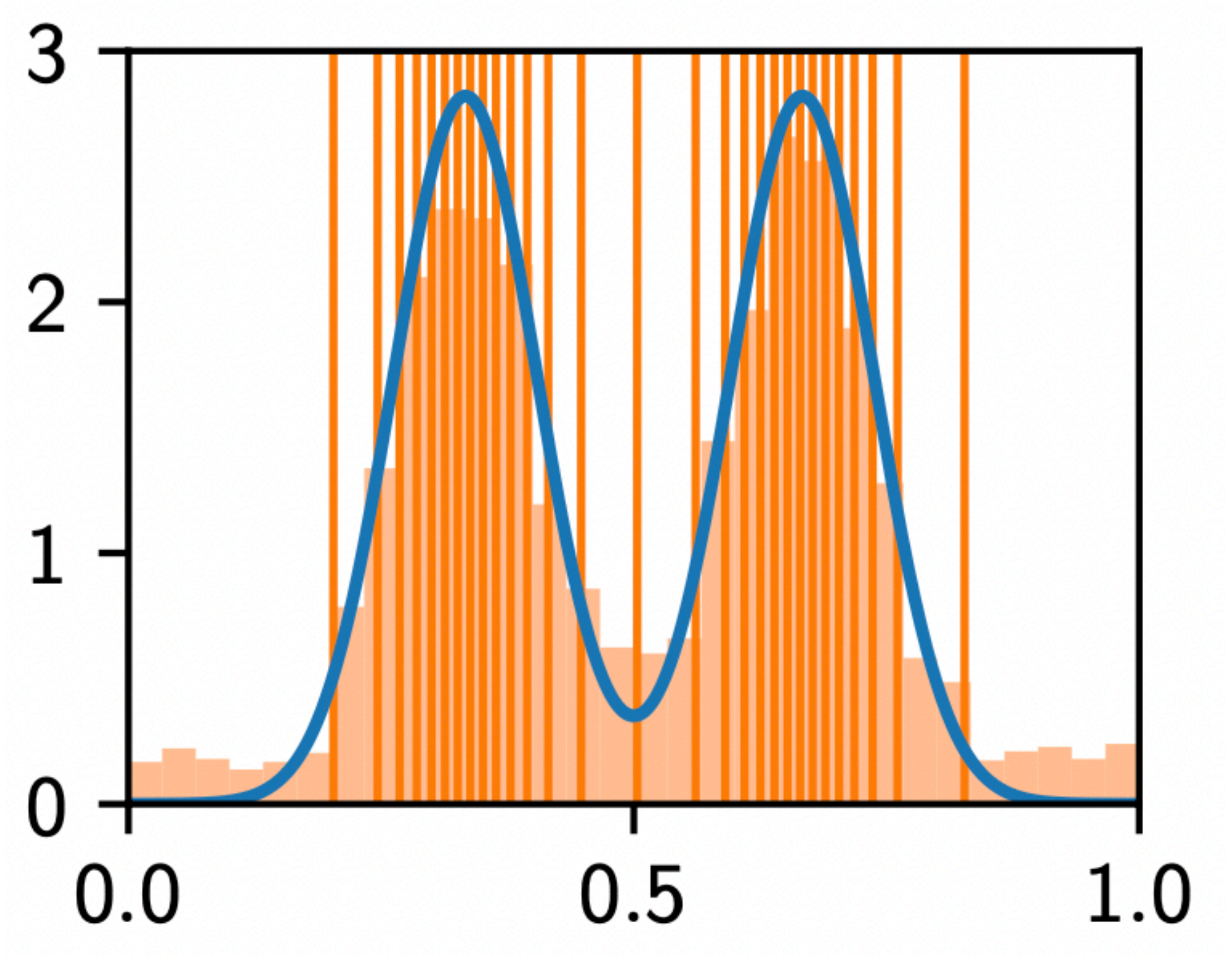

[G. P. Lepage, 1978]

# VEGAS algorithm

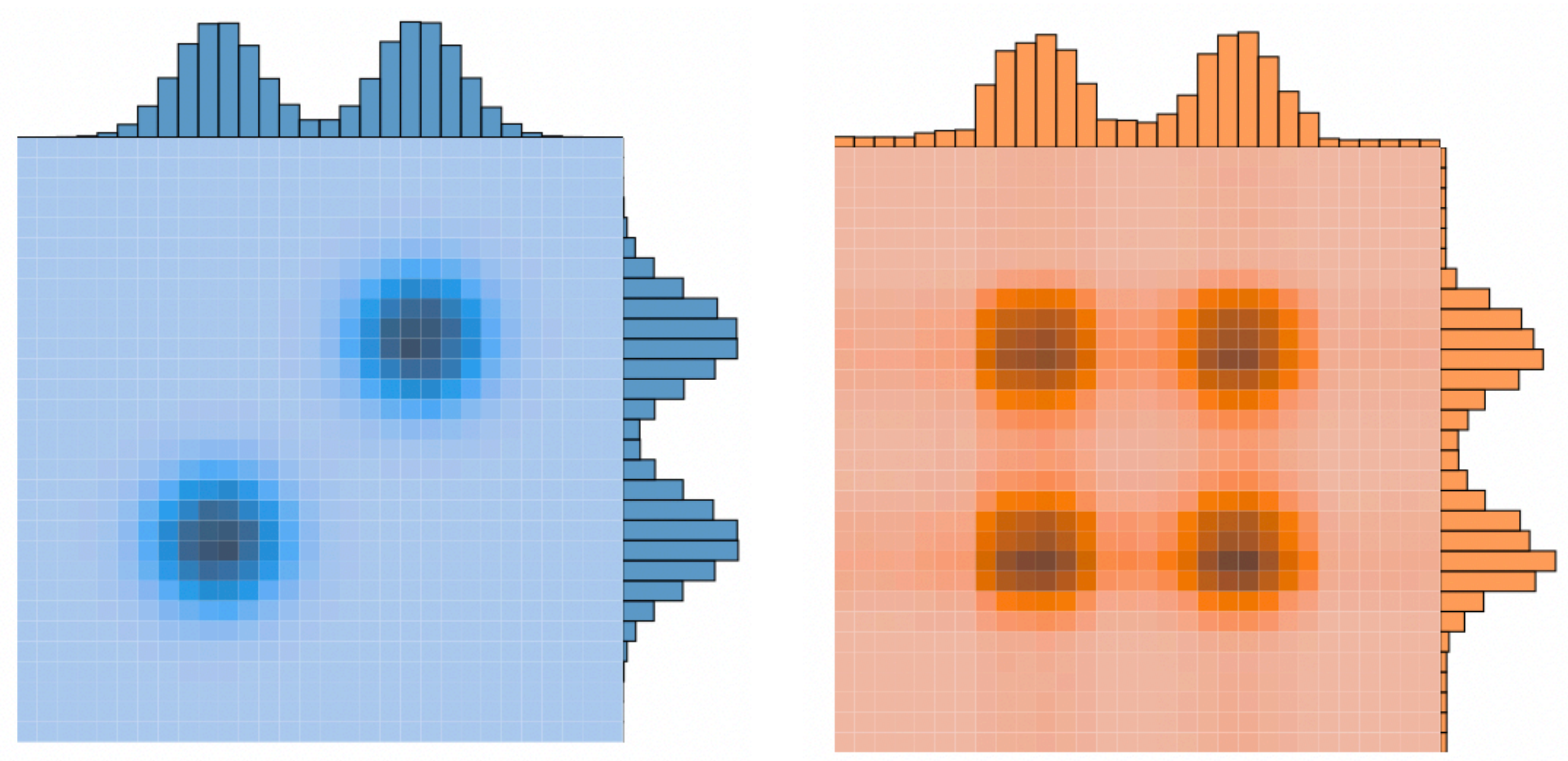
Factorize probability

$$p(x) = p(x_1) \cdots p(x_n)$$

Fit bins with equal probability and varying width




- ⊕ Computationally cheap
- ⊖ High-dim and rich peaking functions → slow convergence
- ⊖ Peaks not aligned with grid axes → phantom peaks



[G. P. Lepage, 1978]

# MadNIS: Neural Importance Sampling

$$I = \sum_i \left\langle \alpha_i(x) \frac{f(x)}{g_i(x)} \right\rangle_{x \sim g_i(x)}$$

$\downarrow$

Use physics knowledge to construct channels and mappings

$\downarrow$ $\downarrow$

Normalizing Flow to refine channel mappings

Fully connected network to refine channel weights

$\downarrow$ $\downarrow$

Optimize simultaneously with integral variance as loss function

# Overview

MadNIS

### Basic functionality

- Neural Channel Weights
- Normalizing Flow
- MadGraph matrix elements
- MadEvent channel mappings

### Improved Multichanneling

- Stratified sampling/ training
- Symmetries between channels
- Removing channels
- Partial weight buffering

### Improved training

- VEGAS initialization
- Buffered training
- Surrogate integrand

# Overview

MadNIS

**Basic functionality**

- Neural Channel Weights
- Normalizing Flow
- MadGraph matrix elements
- MadEvent channel mappings

**Improved Multichanneling**

- Stratified sampling/ training
- Symmetries between channels
- Removing channels
- Partial weight buffering

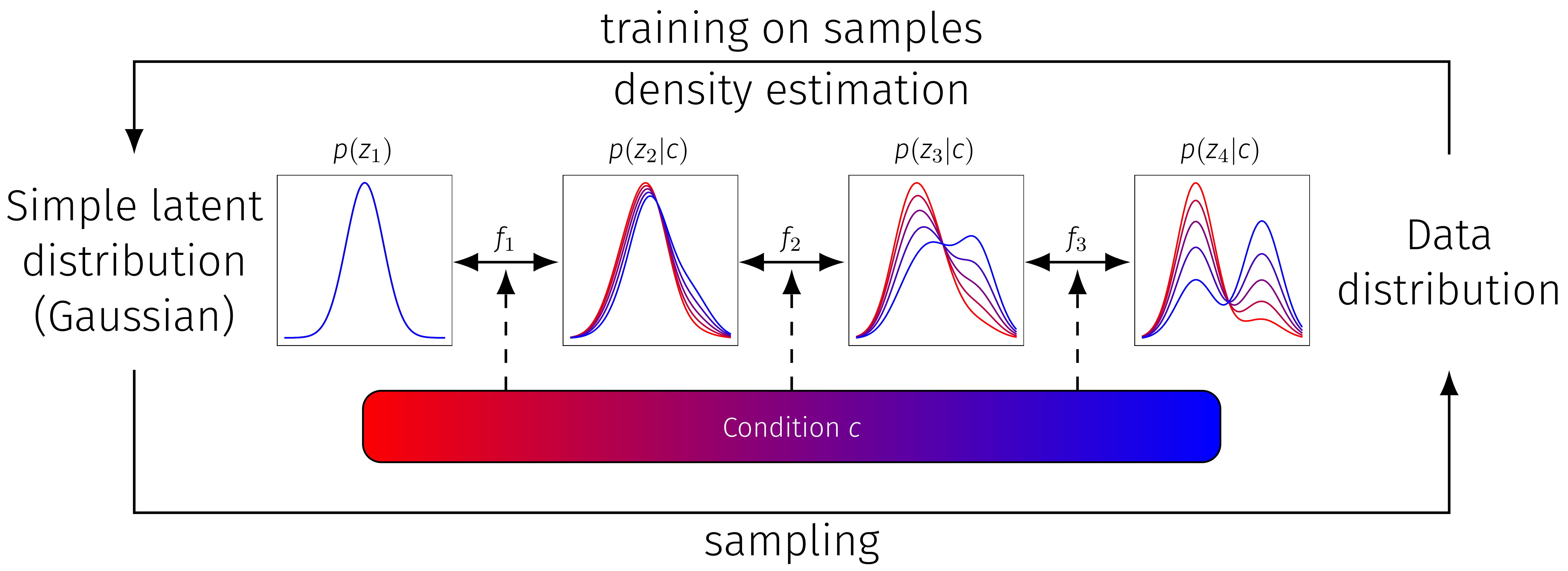
**Improved training**

- VEGAS initialization
- Buffered training
- Surrogate integrand

# Normalizing Flows

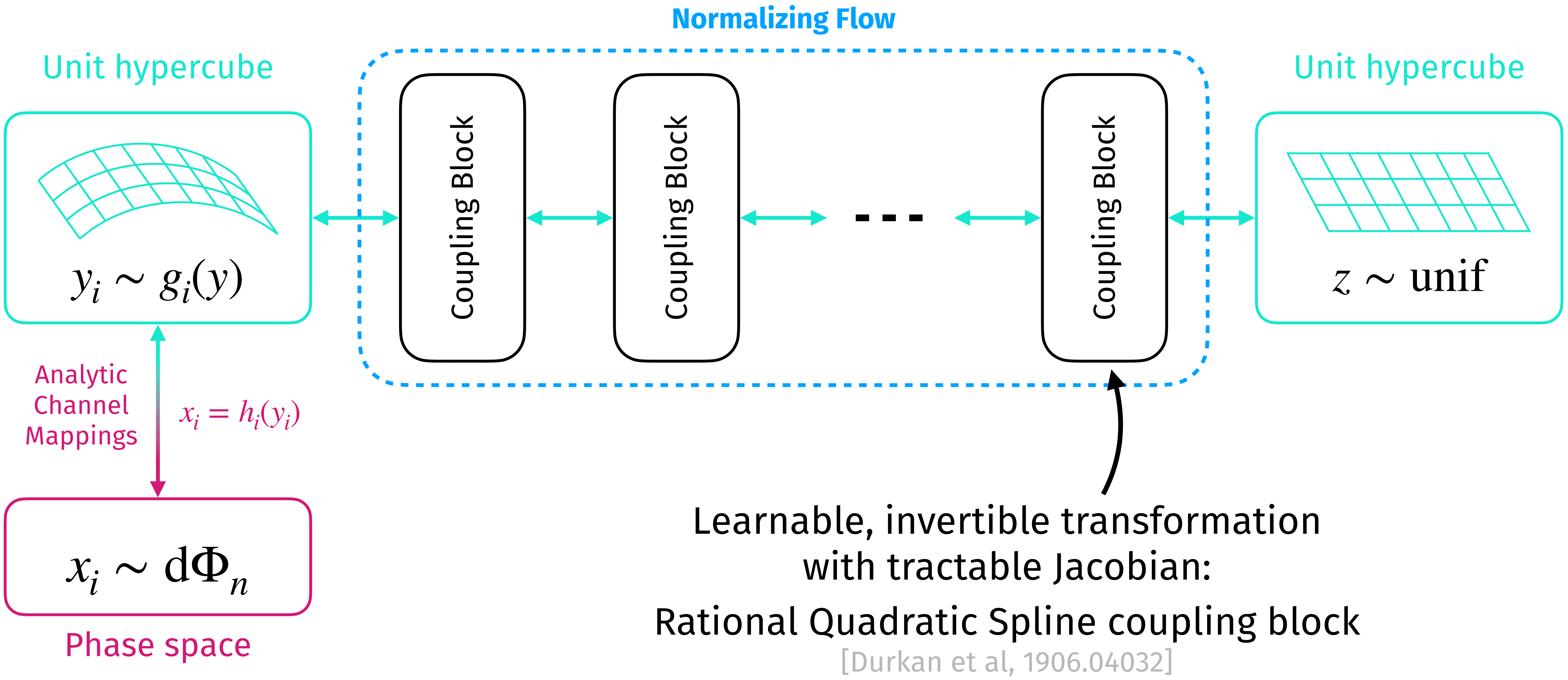
Chain of **invertible**, **learnable** transformations with **exact likelihood** from change of variables formula

$$\log p(z_n \;\; c) = \log p(z_1) + \log \det \frac{\partial z_1(z_n; c)}{\partial z_n}$$




Flows for NIS: [Gao et al, 2001.05486] [Gao et al, 2001.10028] [Bothmann et al, 2001.05478]

# Neural Importance Sampling




Flows for NIS: [Gao et al, 2001.05486] [Gao et al, 2001.10028] [Bothmann et al, 2001.05478]

# Overview

**Basic functionality**

- Neural Channel Weights
- Normalizing Flow
- MadGraph matrix elements
- MadEvent channel mappings

MadNIS

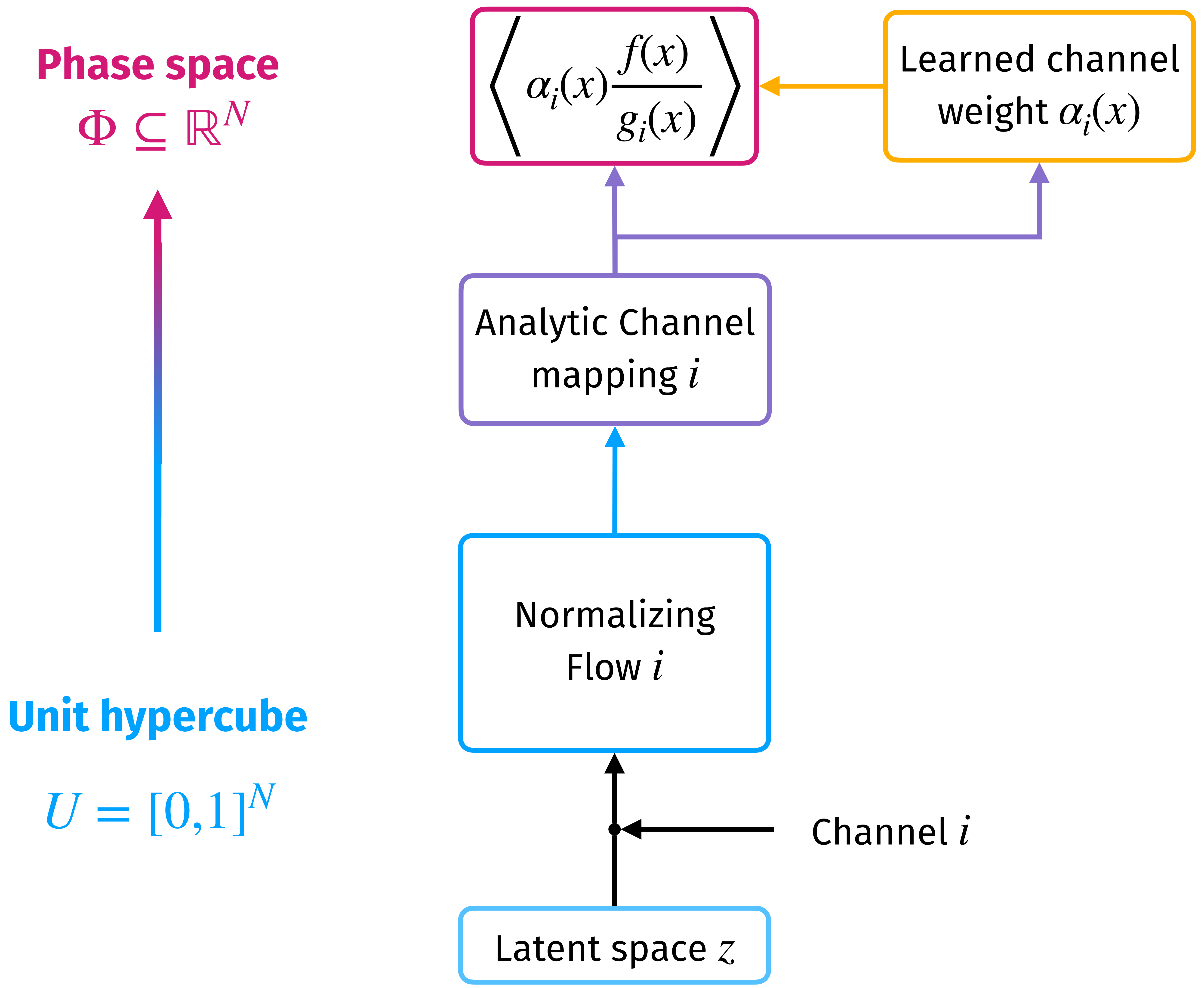
**Improved Multichanneling**

- Stratified sampling/ training
- Symmetries between channels
- Removing channels
- Partial weight buffering

**Improved training**

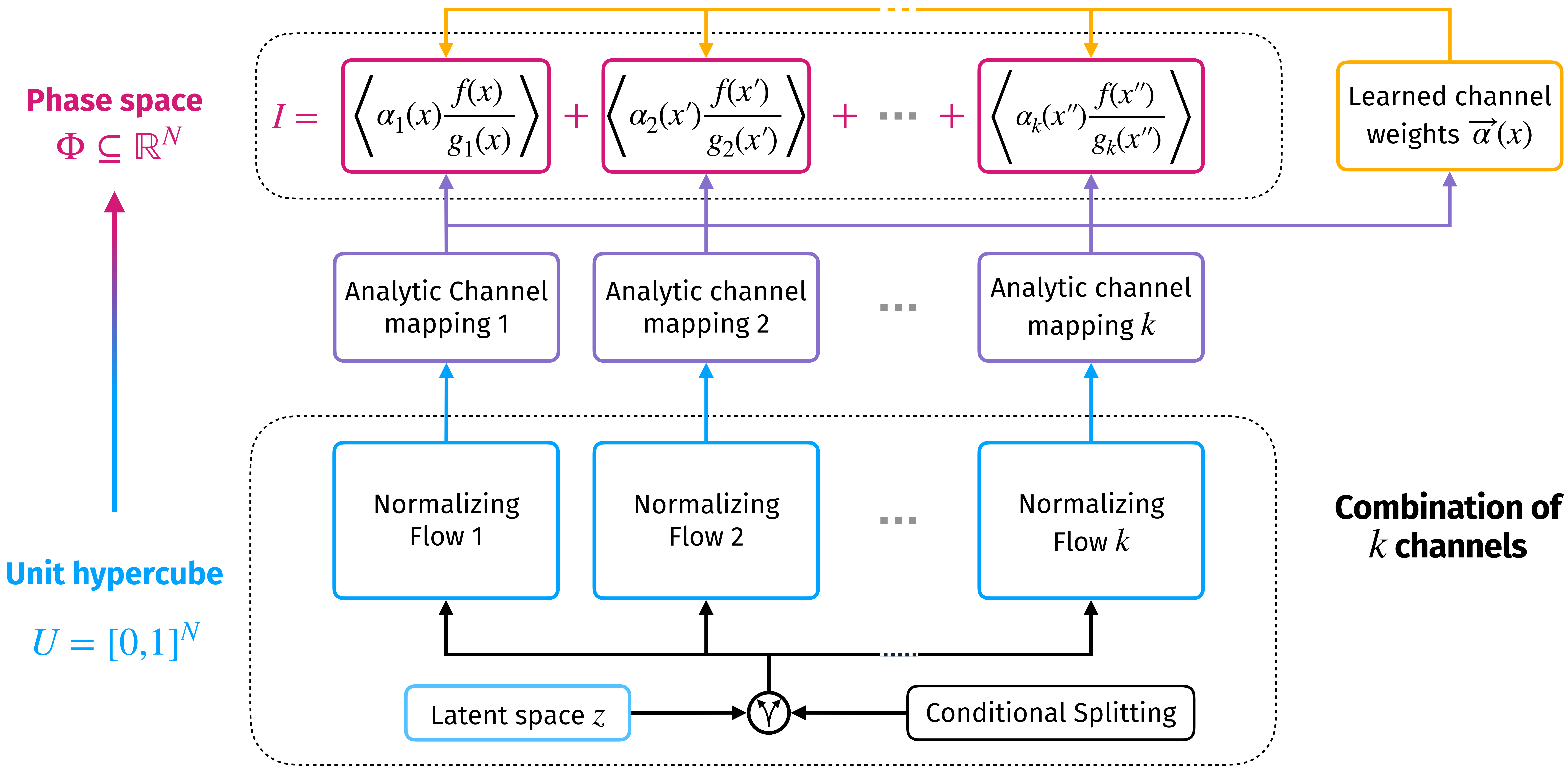
- VEGAS initialization
- Buffered training
- Surrogate integrand

# MadNIS: Neural Importance Sampling

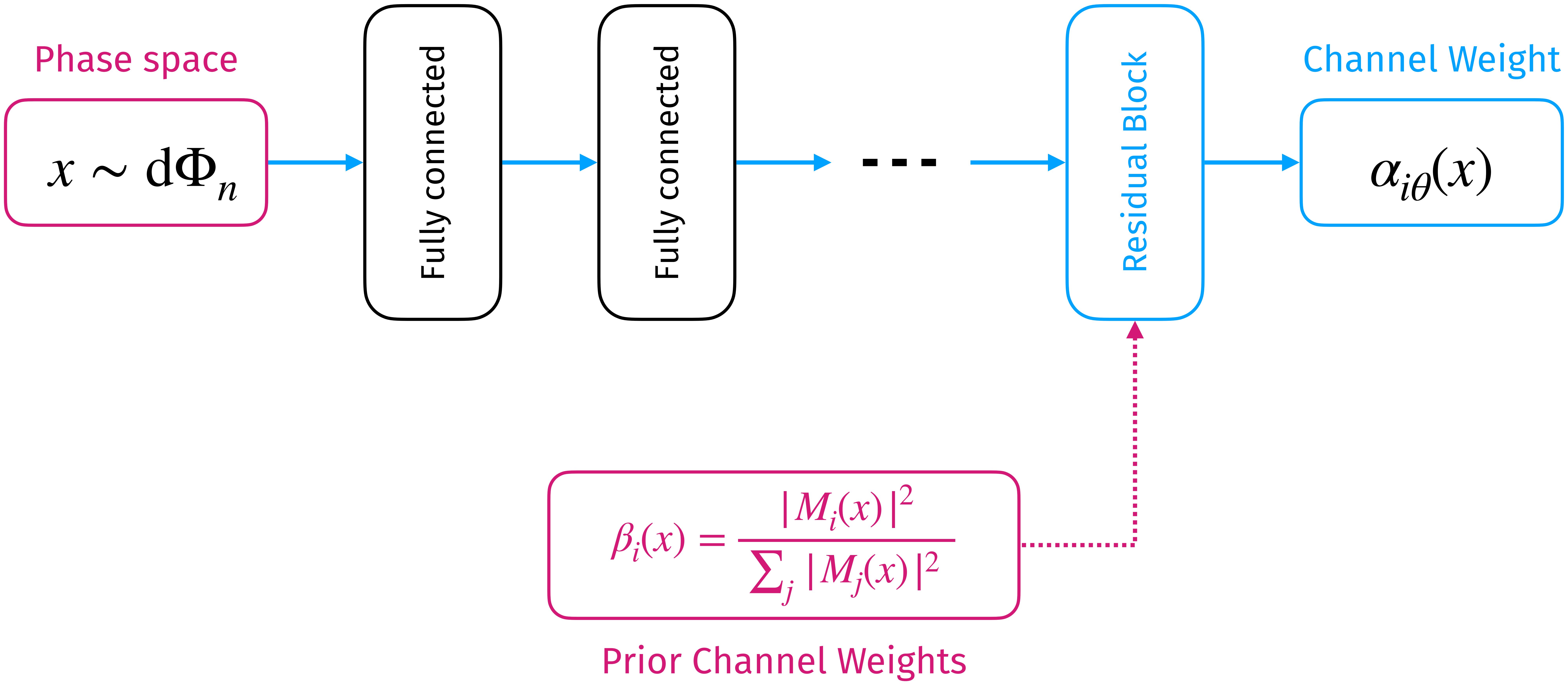



**Single channel $i$**

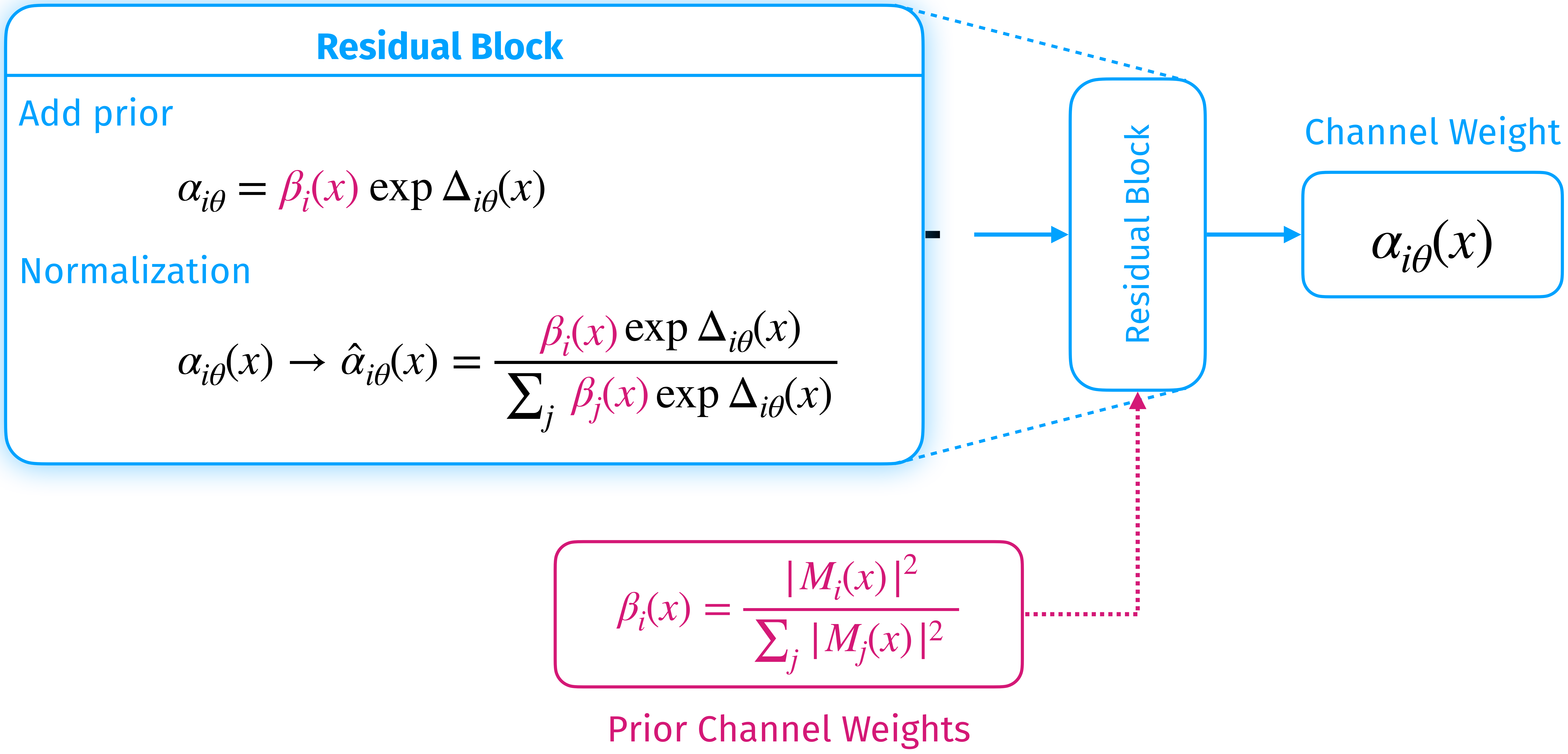
# MadNIS: Neural Importance Sampling



# Neural Channel Weights



# Neural Channel Weights



# Loss function

Training objective:
Minimize total variance

$$\sigma_{\text{tot}}^2 = N \sum_i \frac{\sigma_i^2}{N_i} \quad \text{with}$$

$$\sigma_i^2 = \text{Var}\left( \alpha_i(x) \frac{f(x)}{g_i(x)} \right)_{x \sim g_i(x)}$$

Optimal MC weights depend on $N_i$

↓

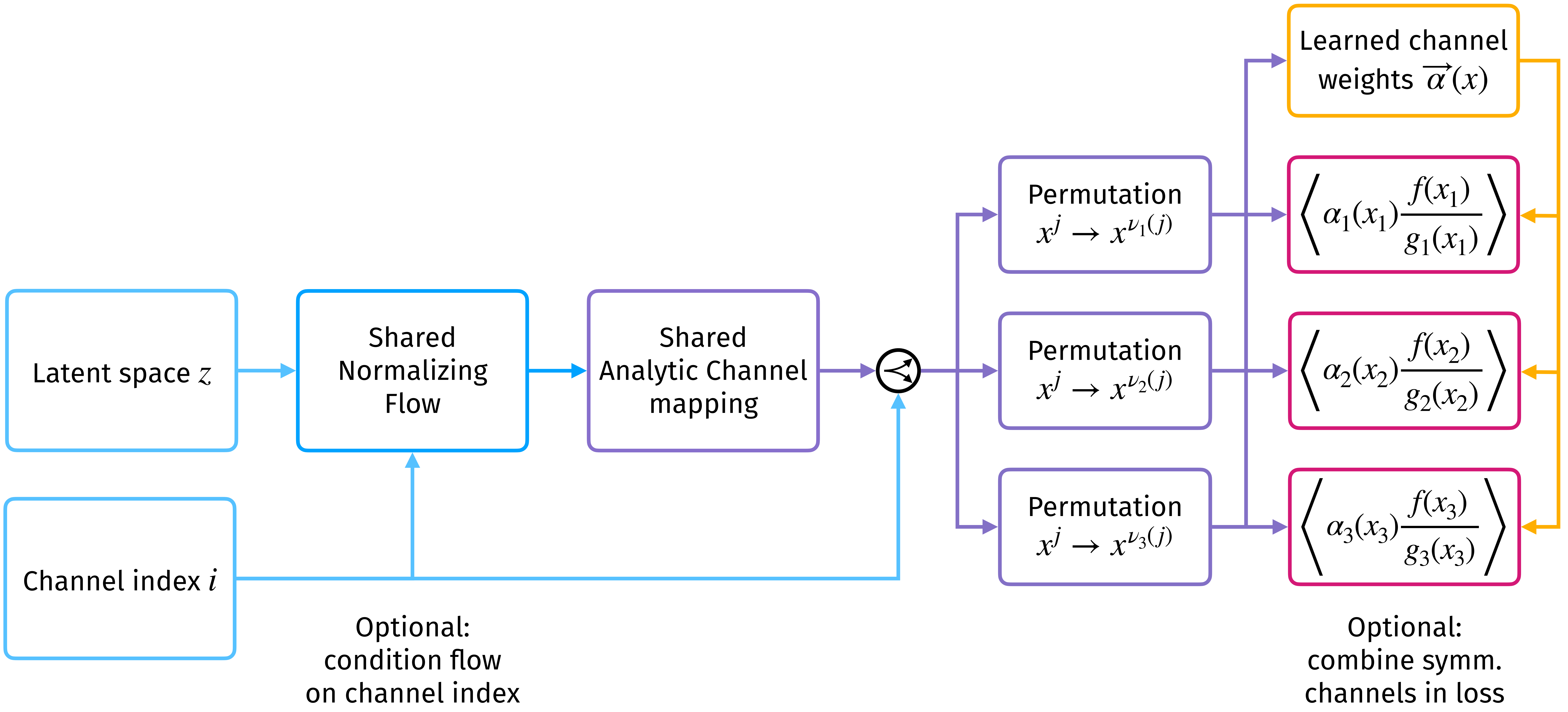
assume choice of $N_i$ during training:
use stratified sampling

$$N_i = N \frac{\sigma_i}{\sum_k \sigma_k}$$

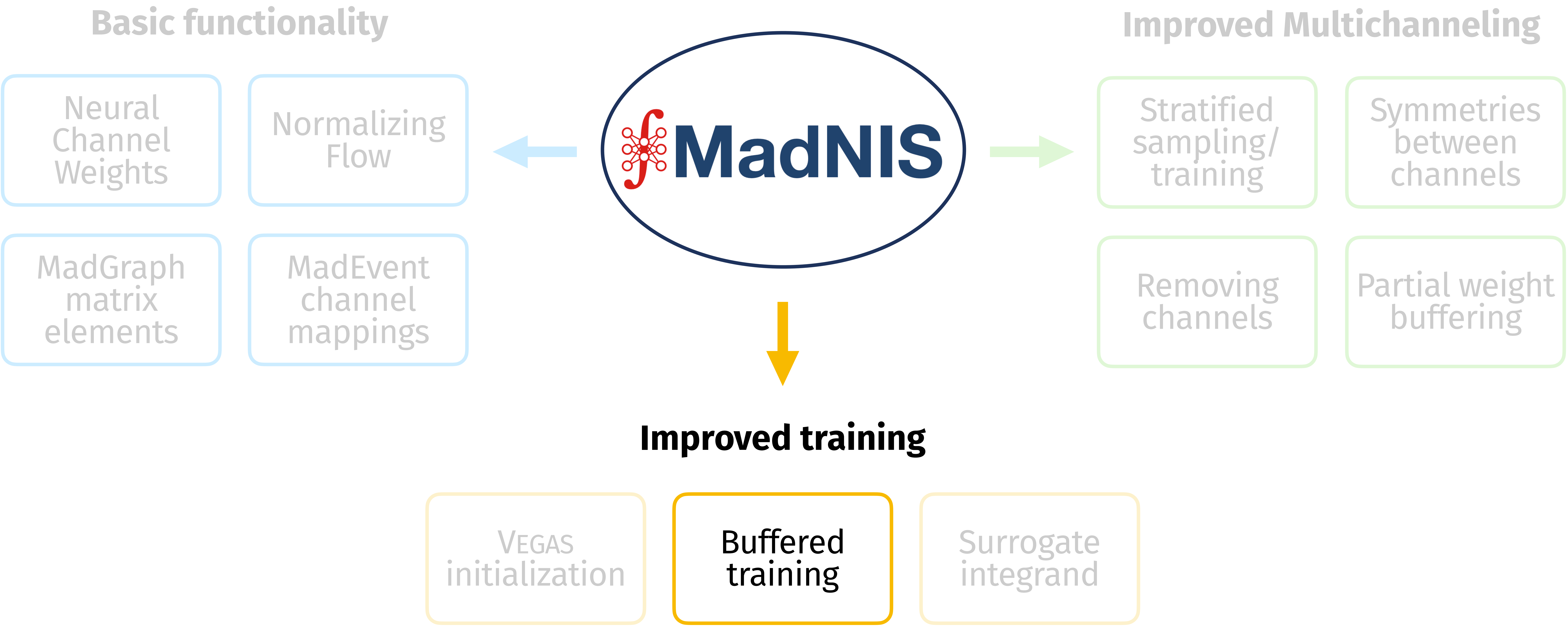
**MadNIS loss function**

$$\mathscr{L} = \sigma_{\text{tot}}^2 = \sum_{i,k} \sigma_i \, \sigma_k$$

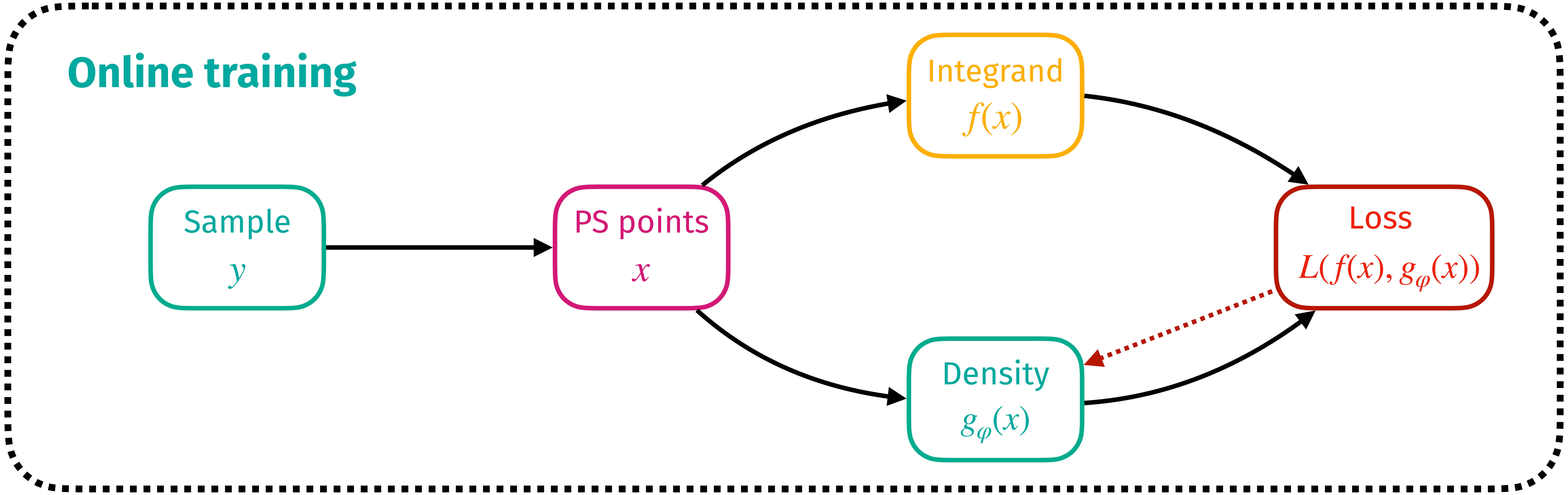
# SymFI Multi-Channel



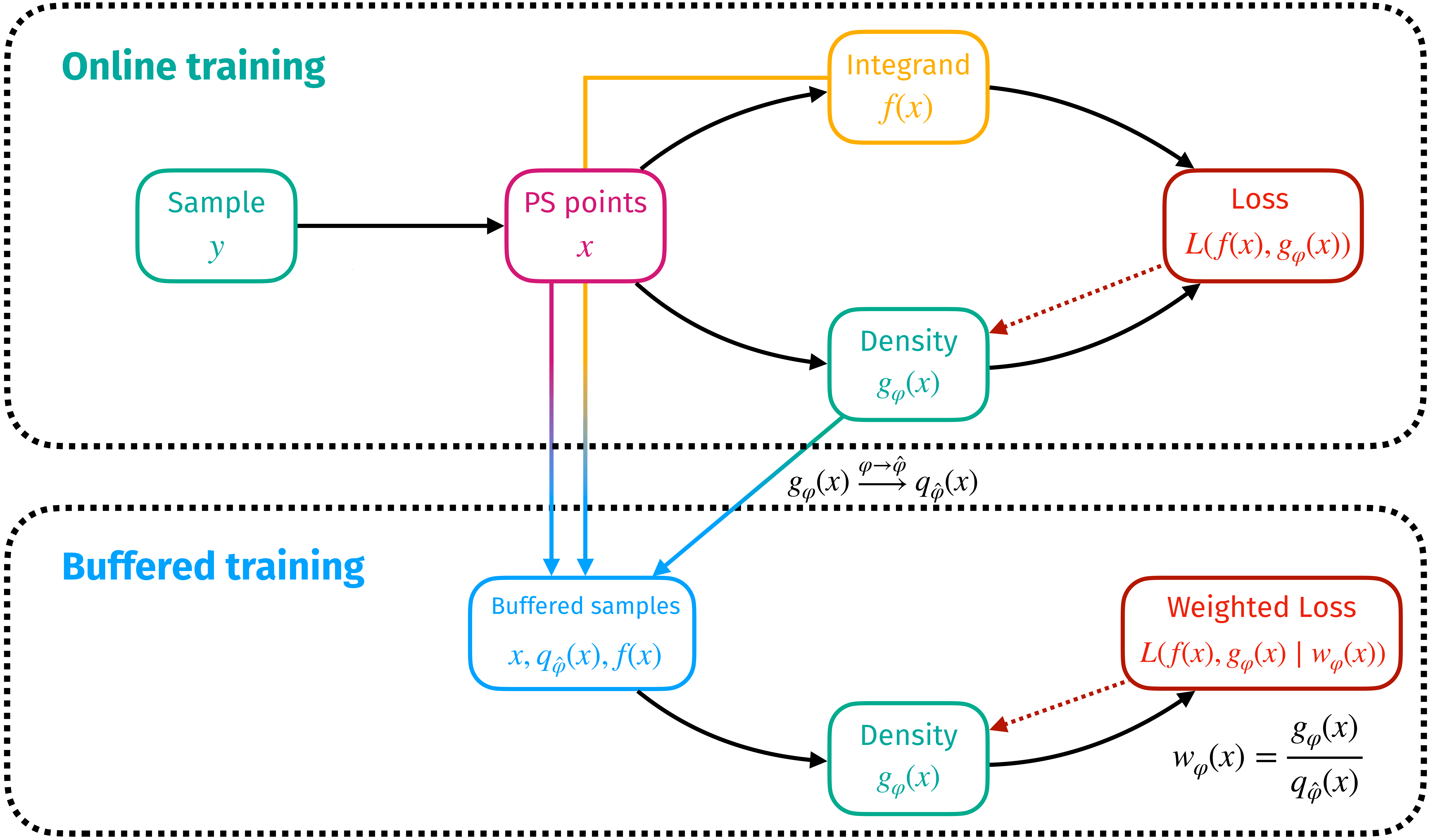
# Overview


Basic functionality
Neural Channel Weights
Normalizing Flow
MadGraph matrix elements
MadEvent channel mappings
MadNIS
Improved Multichanneling
Stratified sampling/ training
Symmetries between channels
Removing channels
Partial weight buffering
Improved training
VEGAS initialization
Buffered training
Surrogate integrand

# Buffered Training



# Buffered Training



# Buffered Training

**Training algorithm**

generate new samples, train on them, save samples

↓

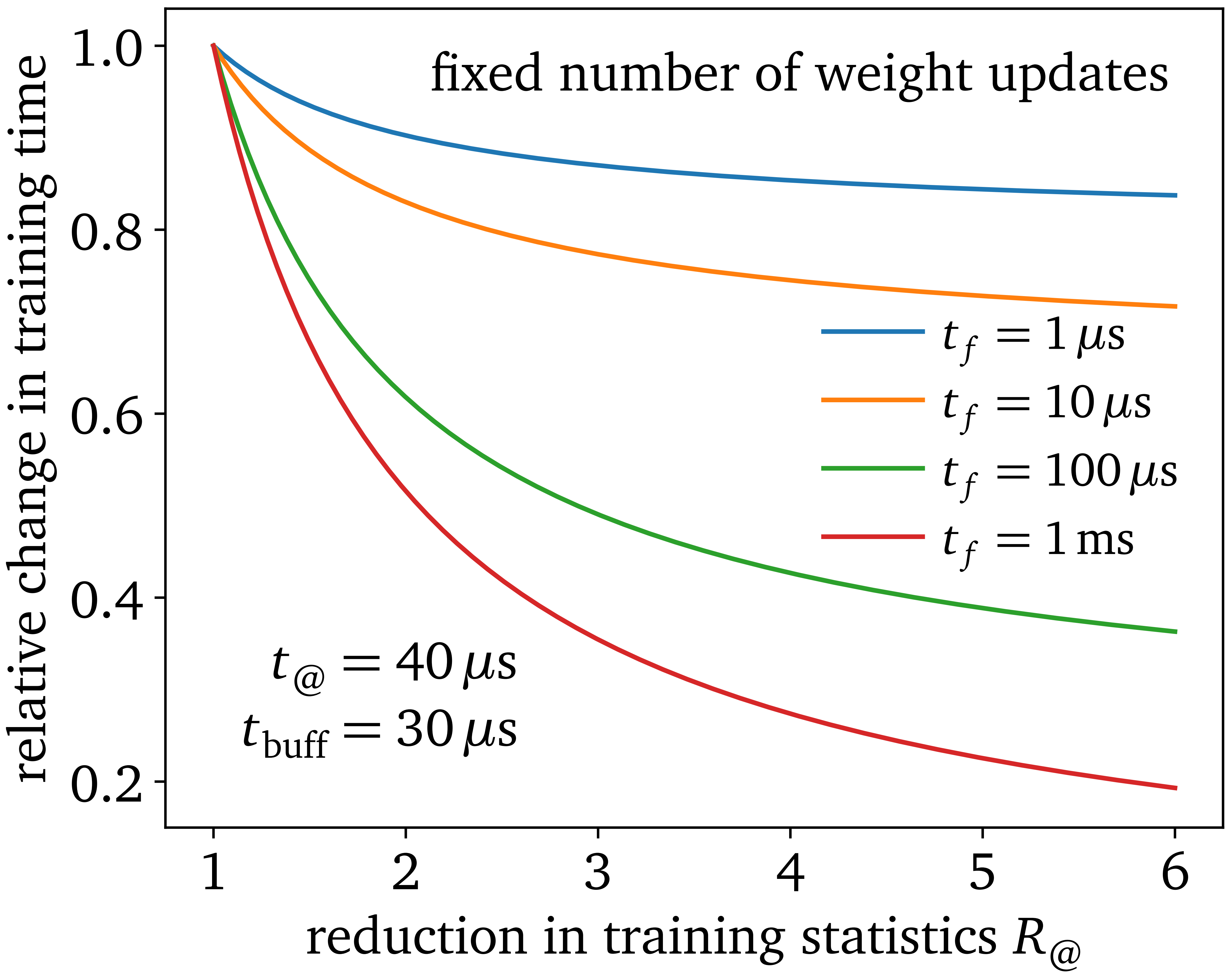
train on saved samples $n$ times

↓

repeat

Reduction in training statistics by

$$R_@ = n + 1$$



# Overview

**Basic functionality**

- Neural Channel Weights
- Normalizing Flow
- MadGraph matrix elements
- MadEvent channel mappings

MadNIS

**Improved Multichanneling**

- Stratified sampling/ training
- Symmetries between channels
- Removing channels
- Partial weight buffering

**Improved training**

- VEGAS initialization
- Buffered training
- Surrogate integrand

# VEGAS Initialization

| | VEGAS | Flow |
|---|---|---|
| Training | Fast | Slow |
| Correlations | No | Yes |

↓

Combine advantages:

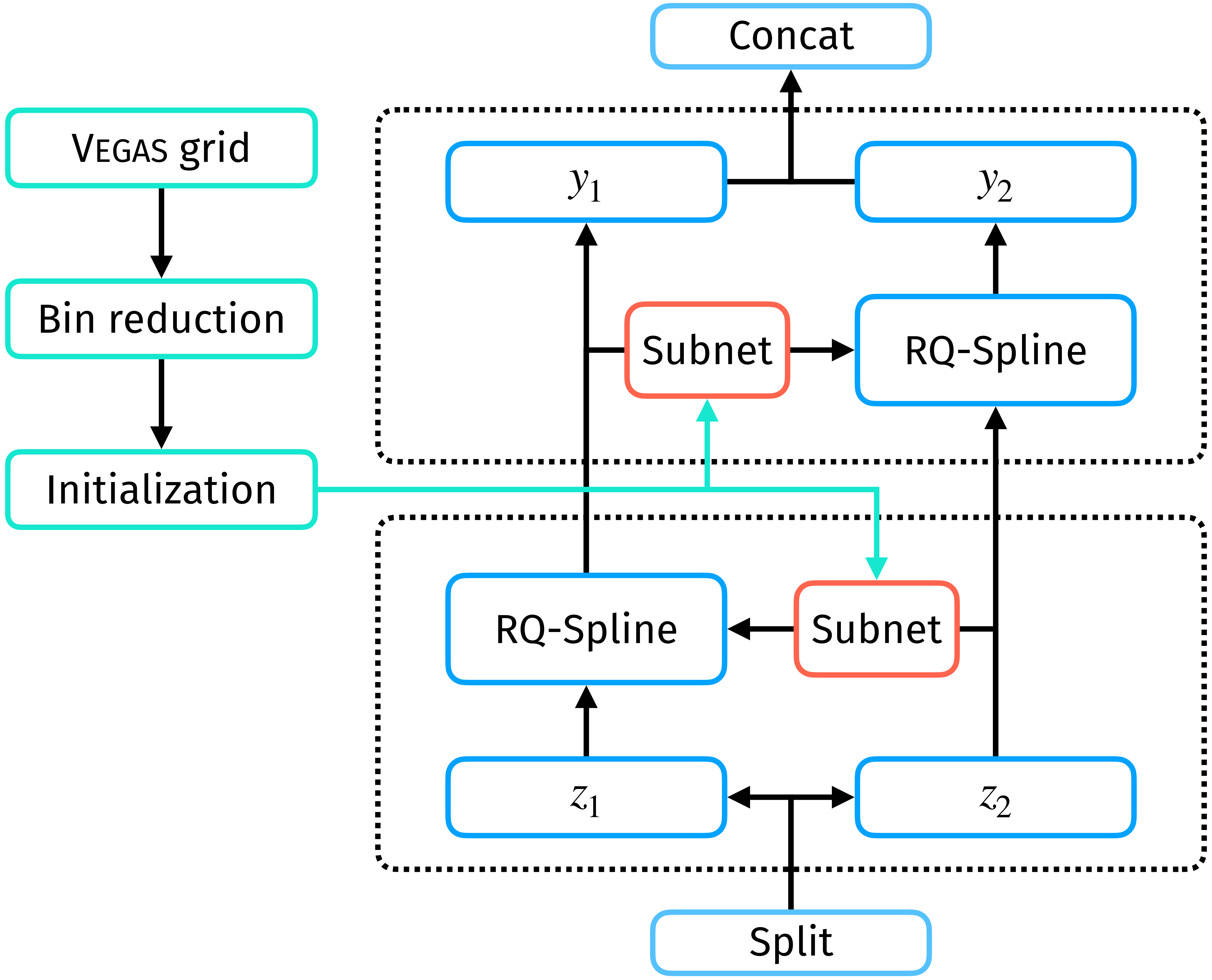
Pre-trained VEGAS grid as starting point for flow training

# VEGAS Initialization

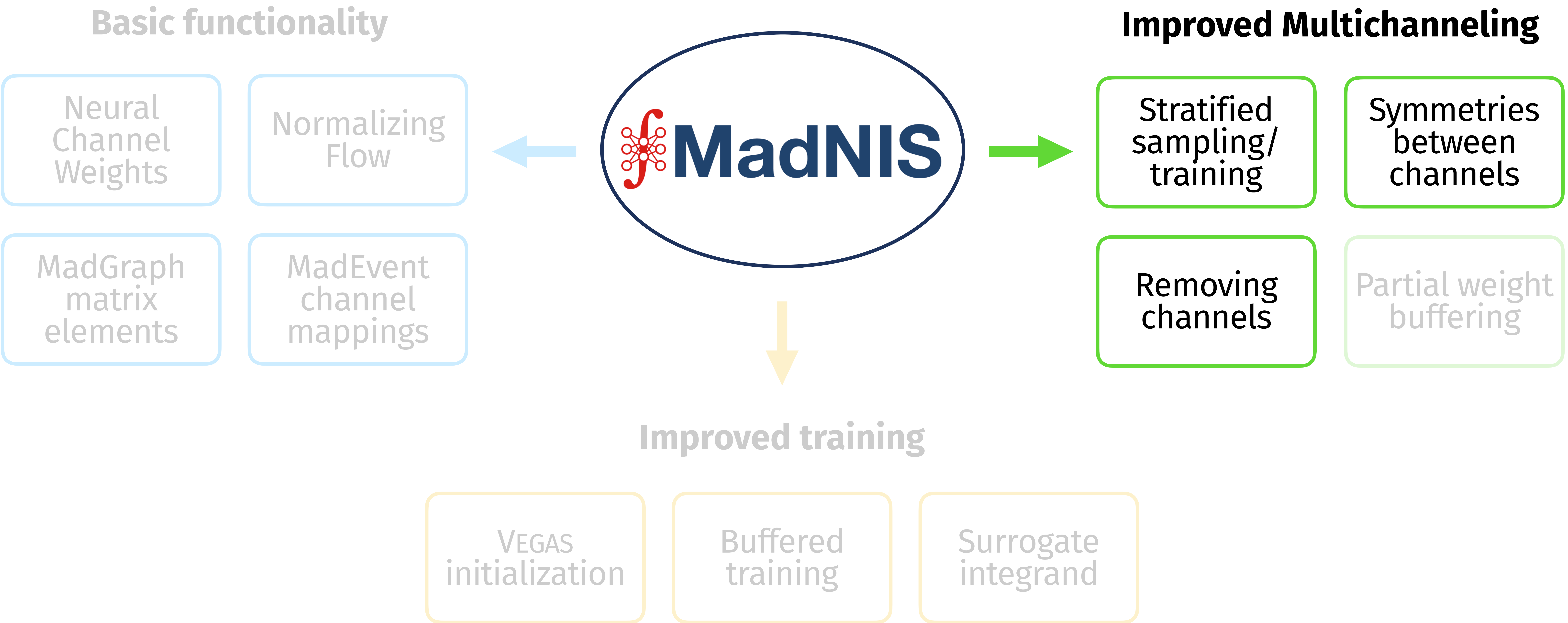
| | VEGAS | Flow |
|---|---|---|
| Training | Fast | Slow |
| Correlations | No | Yes |

Combine advantages:

Pre-trained VEGAS grid as starting point for flow training



# Overview


Basic functionality
Neural Channel Weights
Normalizing Flow
MadGraph matrix elements
MadEvent channel mappings
MadNIS
Improved Multichanneling
Stratified sampling/ training
Symmetries between channels
Removing channels
Partial weight buffering
Improved training
VEGAS initialization
Buffered training
Surrogate integrand

# Improved Multichanneling

**Use symmetries**

Groups of channels only **differ by permutations** of final state momenta

↓

use **common flows** and combine in loss function

**Stratified training**

Channels have different contributions to the total variance

↓

**more samples** for channels with **higher variance** during training

**Channel dropping**

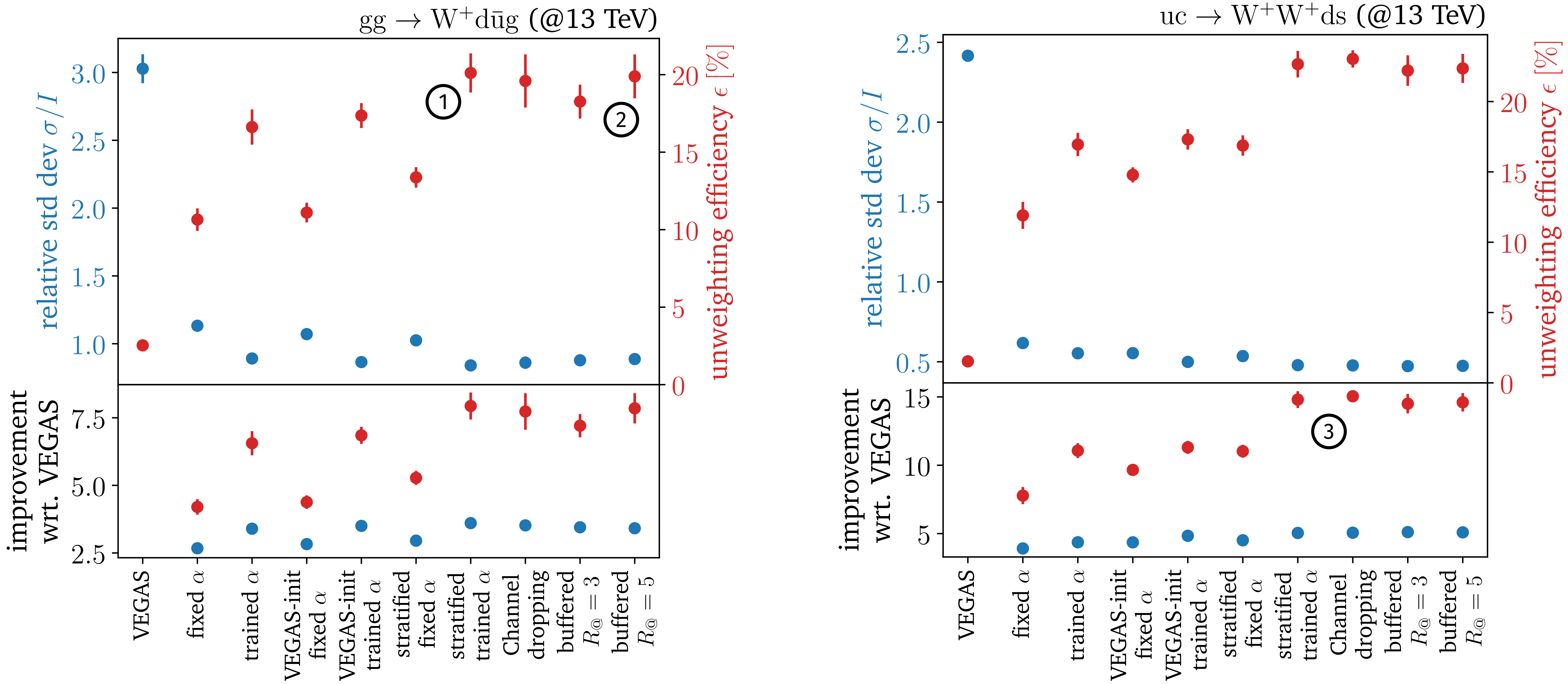
MadNIS often **reduces contribution** of some channels to total integral

↓

**remove** insignificant channels from the training completely

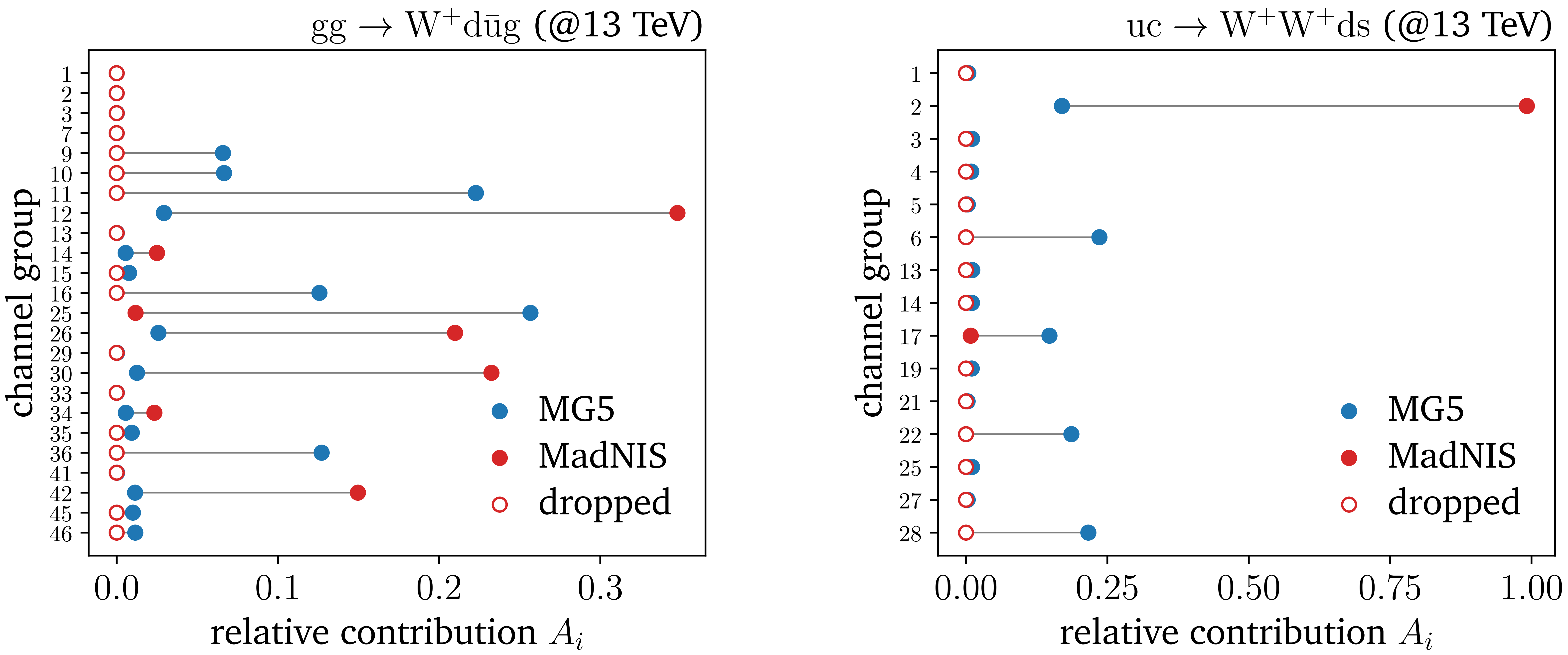
Reduced complexity
Improved stability

# LHC processes




1. Excellent results by combining all improvements!
2. Same performance with buffered training
3. Even larger improvements for process with large interference terms

# Learned channel weights

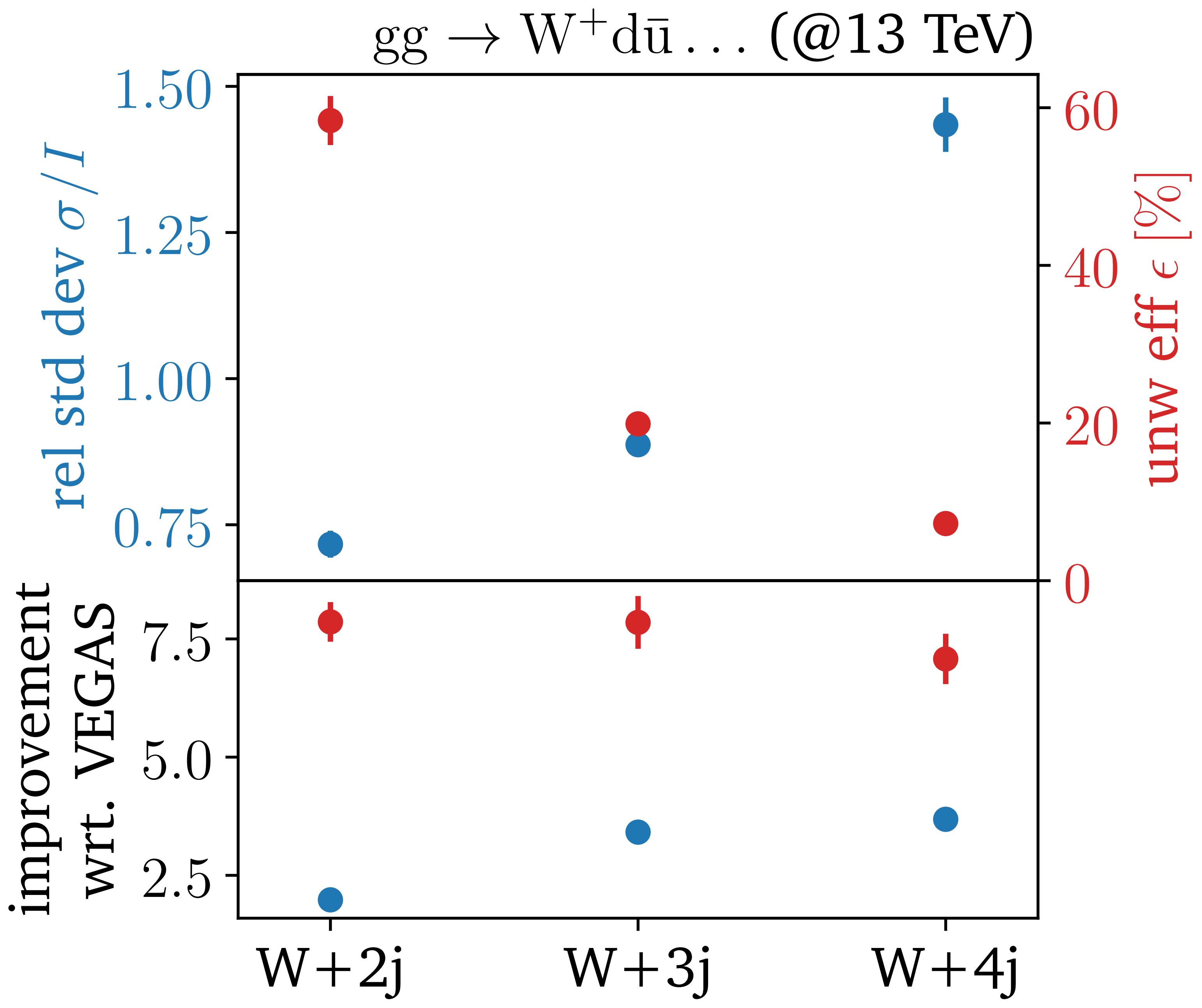



MadNIS often sends weight of many channels to 0

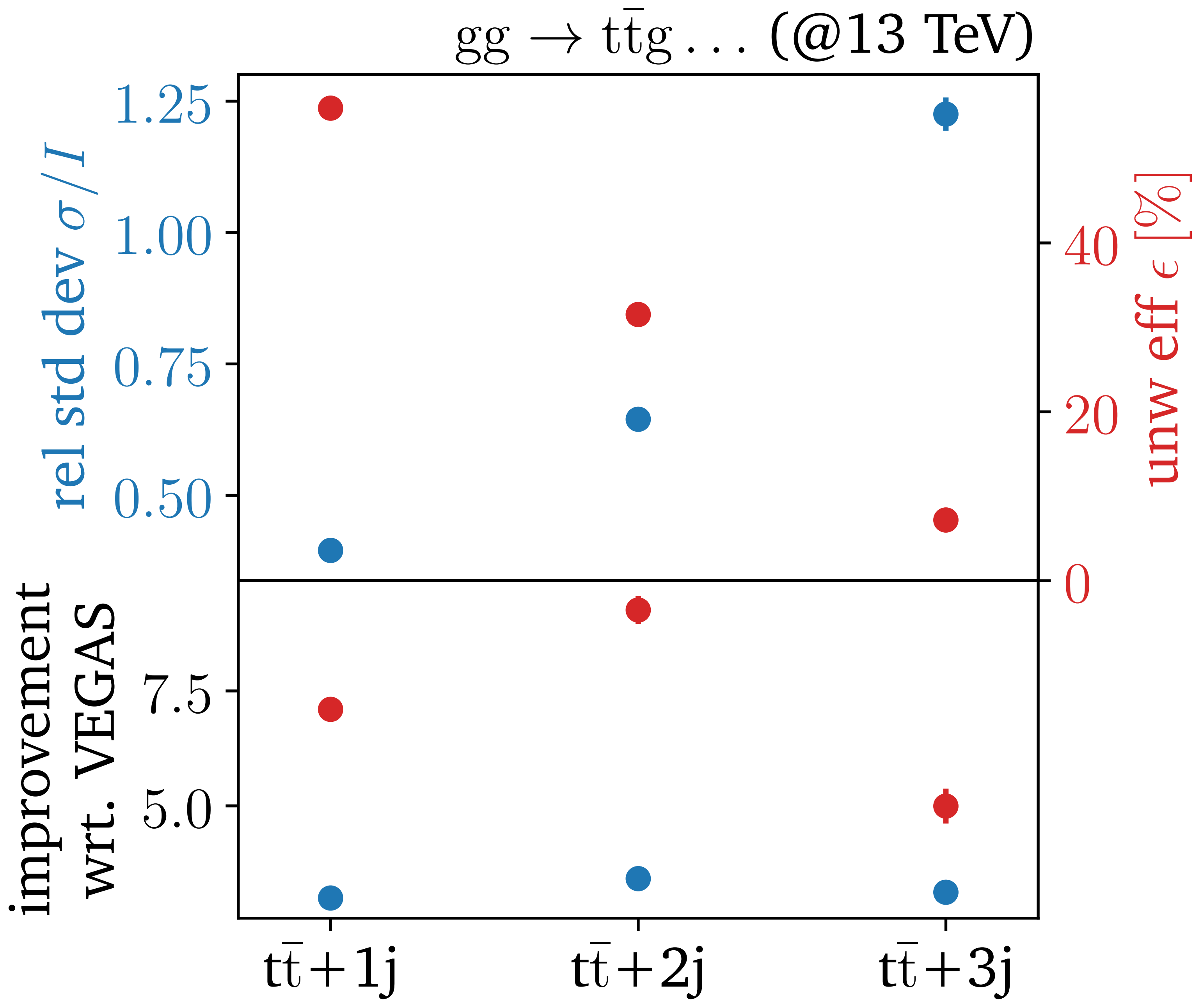
↓

dropping channels makes training and
event generation more stable and efficient

# Scaling with multiplicity




$gg \to W^+ d\bar{u} gg$
384 channels, 108 symm.
7x better than VEGAS




$gg \to t\bar{t}ggg$
945 channels, 119 symm.
5x better than VEGAS

Large improvements compared to VEGAS even for high multiplicities and many channels!

# Outlook

### The MadNIS Reloaded

Large improvements,
even for high multiplicities
and complicated processes!

[2311.01548]

### Future plans

Make MadNIS part of next
MadGraph version