MOMENT MORPHING$^a$

Interpolation between multi-dimensional histograms

Max Baak$^1$, Stefan Gadatsch$^2$, Robert Harrington$^3$, Wouter Verkerke$^4$

16th April, 2015

$^1$CERN, $^2,^4$Nikhef, $^3$University of Edinburgh

$^a$M. Baak, S. Gadatsch, R. Harrington and W. Verkerke 2014.
Given a set of measurements $x$, and a hypothesis $\alpha$, what is the probability of the data under this hypothesis?

Define a probability model $f(x|\alpha)$.

Generally no analytical predictions for $f(x|\alpha)$ available at LHC.

Monte Carlo simulation (CPU expensive and time consuming) for “reference” parameter values.

Statistical inference requires maximizing the likelihood.

Likelihood function must be defined for all values of $\alpha$.

Interpolate between available samples.

Assumption: resolution is larger than simulated step size.
Changing **mean**

E.g. shifting mass peak, etc.

Changing **variance**

E.g. resonance width, etc.

In the following construct a continuous parametric model for all other parameter values
 STEP 1 – VERTICAL MORPHING

- Suppose $f(x|m)$ is known for $n$ values of $m$

- Write Taylor series up to order $n – 1$ around a reference value $m_0$

$$f(x|m_i) \approx \sum_{j=0}^{n-1} (m_i - m_0)^j \frac{1}{j!} \frac{d^j f(x|m_0)}{dm^j} \quad \Leftrightarrow \quad f'_j(x|m_0) = \sum_{i=0}^{n-1} (M^{-1})_{ji} f(x|m_i)$$

- Predicted template shape at any value $m'$ is a linear combination of the reference templates (“vertical” morphing)

$$f_{\text{pred}}(x|m') = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (m' - m_0)^j (M^{-1})_{ji} f(x|m_i)$$

$$= c_i(m')$$

- Coefficients $c_i$ add up to 1
A TOY EXAMPLE

morphing parameter

Projection of p.d.f.s

0 0.5 1

0 0.02 0.04 0.06 0.08 0.1 0.12 0.14

1st input template (mapped to m=0)
2nd input template (mapped to m=1)
Vertical morphing interpolation (at m=0.5)
Linear transformation of the input observables $f(x|m_i) \rightarrow f(x'|m_i)$

Translating referred to as "horizontal" morphing

$$x'_{ij} = (x_j - \mu'_j) \frac{\sigma_{ij}}{\sigma'_j} + \mu_{ij}$$

$$\mu'_j(m) = \sum_i c_i(m) \cdot \mu_{ij}$$

with

$$\sigma'_j(m) = \sum_i c_i(m) \cdot \sigma_{ij}$$

Normalization only changed by the Jacobian

$$\int_{-\infty}^{+\infty} f(x'|m_i)dx = \frac{1}{\prod_j a_j(m')} \int_{-\infty}^{+\infty} f_i(x|m_i)dx$$

Final p.d.f. is self-normalized

$$p(x|m') = \sum_i c_i(m')f(x', m_i) \prod_j a_j(m')$$
BACK TO THE TOY EXAMPLE

morphing parameter

0 0.5 1

Projection of p.d.f.s

0 50 100 150 200 250 300 350 400

1\textsuperscript{st} input template (mapped to m=0)
2\textsuperscript{nd} input template (mapped to m=1)
Vertical morphing interpolation (at m=0.5)
MomentMorph interpolation (at m=0.5)
(accounting only for shifted mean)
// This example builds two normal distributions and uses moment morphing to
// interpolate between the templates using RooFit.
using namespace RooFit;

// Create a persistable container for RooFit projects, allowing to use a simplified
// scripting language to build the p.d.f.s needed in this example
RooWorkspace w("w", 1);

// Build two normal distributions, corresponding to different values in the morph
// parameter space. They share the same observable, but have otherwise different
// moments, i.e. mean and width.
w.factory("RooGaussian::gaussian1(obs[0,400],50,15)");
w.factory("RooGaussian::gaussian2(obs,300,40)");

// Build a RooMomentMorph p.d.f. which interpolates between the normal distributions
// created before. The interpolation is parametrized by the parameter alpha and the
// reference templates map to alpha=0 and alpha=1 respectively.
w.factory("RooMomentMorph::morphpdf(alpha[0,1],obs,{gaussian1,gaussian2},{0,1})");

// Set the morphing parameter alpha explicitly to 0.5.
w::alpha->setVal(0.5);

// Create a frame to draw the p.d.f. from before and show the input templates as
// solid blue curves and the moment morph p.d.f. at alpha=0.5 in dashed red.
RooPlot* frame = w::obs->frame();
w::gaussian1->plotOn(frame, LineColor(kBlue), LineStyle(kSolid));
w::gaussian2->plotOn(frame, LineColor(kBlue), LineStyle(kSolid));
w::morphpdf->plotOn(frame, LineColor(kRed), LineStyle(kDashed));
frame->Draw();
More Examples

Cauchy distribution \(\xrightarrow{\text{\quad}}\) Crystal Ball
line shape \(\xrightarrow{\text{\quad}}\) normal distribution

Standard Model Higgs boson
decaying to four leptons
Variations around a nominal distribution
- Typically just the nominal distribution and ±1σ variations simulated

\[ f(x|\theta)|_{\theta=0, \pm 1} \rightarrow f(x|\theta) \text{ for all } \theta \]

Assume that systematic effects in physics measurement factorize: \( L(\alpha_i, \alpha_j) = L(\alpha_i)L(\alpha_j) \)

- Used for background description in ATLAS
  \( H \rightarrow ZZ^* \rightarrow 4\ell \) analysis
- Illustration for three uncertainties
  - Flat, low \( m_{4\ell} \), high \( m_{4\ell} \)
Extendable to $M \geq 1$ observables and $N \geq 1$ model parameters

Reference points on a $N$-dimensional hypercube

Model the impact of a non-factorizable response: $L(\alpha_i, \alpha_j) \neq L(\alpha_i)L(\alpha_j)$

E.g. $b$-tagging calibration depends (somewhat) on jet calibration

```cpp
1 // Define axes that span a hypercube
2 RooBinning dim1(1, 0.0, 1.0);
3 RooBinning dim2(1, 0.0, 1.0);

// Attach reference distributions to bin boundaries
5 RooMomentMorphND::Grid referenceGrid(dim1, dim2);
6 referenceGrid.addPdf(*gauss1, 0, 0);
7 referenceGrid.addPdf(*gauss3, 1, 0);
8 referenceGrid.addPdf(*gauss2, 0, 1);
9 referenceGrid.addPdf(*gauss4, 1, 1);
```

Subject to changes in the covariance moments
AN EXAMPLE IN TWO DIMENSIONS

observable 1
5
−0
5
10
15

observable 2
6
−4
−2
−0
2
4
6
8
10

morphing parameter 2
0
0.5
1

morphing parameter 1
0
0.5
1
1.5

Input templates
True template
Moment morphing
Vertical morphing
68% CL
95% CL
Exact for any distribution with linearly changing first and second moments and fixed higher-order moments.

Comparison of moment morphing\(^a\) with vertical morphing\(^b\) and integral morphing\(^c\) using the Kolmogorov-Smirnov distance between interpolated and true shape as metric.

\(^a\)M. Baak, S. Gadatsch, R. Harrington and W. Verkerke 2014.
\(^b\)E.g. K. Cranmer, G. Lewis, L. Moneta, A. Shibata, and W. Verkerke 2012, and Ref. \(^a\).
\(^c\)A.L. Read 1999.
Deviations from exact prediction due to

- (Local) non-linearity of the first and second moment
  - Approximate by piece-wise linear interpolation or higher order polynomial

- Any dependence on higher moments
  - Empirically accounted for, but accuracy depends on distributions
- Numerically stable and fast
  - Caching of expensive components (numerically computed moments, ...)
  - No need to compute normalization of p.d.f.

- Implementation available in the **RooFit** models library in **ROOT**
Higgs boson rate in $H \rightarrow WW^*$

Simulated samples available in 5 GeV in $m_H$

**ATLAS**

$H \rightarrow WW^* \rightarrow l\nu l\nu$

- $\sqrt{s} = 7$ TeV, 4.5 fb$^{-1}$
- $\sqrt{s} = 8$ TeV, 20.3 fb$^{-1}$

*Obs (\(m_H = 128, \mu = 0.94\))

- Obs ± 1σ
- Obs ± 2σ
- Obs ± 3σ

Higgs boson $CP$ measurement

Combination of 1d ($H \rightarrow WW^*$) and 2d ($H \rightarrow ZZ^*$) moment morphing

$\kappa_{HV}/\kappa_{SM}$

New algorithm to interpolate between Monte Carlo sample distributions

- Generic, simple, fast, stable, accurate
  - Binned histogram and continuous templates in arbitrary dimensions
  - Horizontal and vertical morphing
  - Self-normalized, caching of expensive objects
  - Exact or nearly exact for many typical applications
Backup


Two input distributions $f_1(x)$ and $f_2(x)$ with cumulative distribution functions

$$F_1(x) = \int_{-\infty}^{x} f_1(x')dx' \quad \text{and} \quad F_2(x) = \int_{-\infty}^{x} f_2(x')dx'$$

Find $x_1$ and $x_2$ for which

$$F_1(x_1) = F_2(x_2) = y \equiv \bar{F}(x) \quad \text{with} \quad x = ax_1 + bx_2$$

Interpolated distribution is

$$\bar{F}^{-1}(y) = a\bar{F}_1^{-1}(y) + b\bar{F}_2^{-1}(y) \quad \Leftrightarrow \quad \bar{f}(x) = \frac{f_1(x_1)f_2(x_2)}{af_2(x_2) + bf_1(x_1)}$$