Adding the gradient of the likelihood GUNDAM

clark.mcgrew

September 2026

1 Introduction

As of version 2.x.x, GUNDAM will efficiently reweight events to calculate the likelihood as a function of the fit parameters. However, if we can also calculate the gradient of the likelihood, several powerful, and efficient, techniques for analyzing the likelhood will become available (e.g. direct gradient descent methods, various ANN techniques, and Bayesian integration techniques such as Hamiltonian MC integration). This note does the math necessary to calculate the gradient in terms of stuff that can be directly calculated in GUNDAM. The only piece that is missing (as of right now) is the calculation of the gradient for each dial (however, there are PRs from Adrien that add the interface for the dial gradients)

2 Calculating the gradient of the likelihood

The likelihood is

$$\mathcal{L} = \mathcal{L}_{stat} + \mathcal{L}_{pen} \tag{1}$$

where $\mathcal{L}_{stat}$ statistical term and $\mathcal{L}_{pen}$ is penalty term.

The statistical term is the sum over all the bins, b of the the joint likelihoods for finding the data, D_b, given the expectation, W_b

$$\mathcal{L}_{stat} = \sum_b J(D_b, W_b) \tag{2}$$

$$\mathcal{L}_{pen} = \vec{p}\mathbf{V}^{-1}\vec{p} \tag{3}$$

where the statistical term is

$$\mathcal{L}_{stat} = \sum_b J(D_b, W_b) = \sum_b \mathcal{L}_b \tag{4}$$

with $\mathcal{L}_b$ being defined as the joint likelihood of the data and expectation, $J(D_b, W_b)$, in bin b.

The penalty term

$$\mathcal{L}_{pen} = (\vec{p} - \vec{p}_o)\mathbf{V}^{-1}(\vec{p} - \vec{p}_o) \tag{5}$$

where $\vec{p}$ are the parameters of the likelihood, and $\vec{p}_o$ is the pre-fit expectation. The covariance for the prior expectation is $\mathbf{V}$ which is inverted to find the error matrix.

The gradient of the likelihood is

$$\partial_{\vec{p}}\mathcal{L} = \partial_{\vec{p}}\mathcal{L}_{stat} + \partial_{\vec{p}}\mathcal{L}_{pen} \tag{6}$$

The gradient of the penalty term can be expressed as

$$\partial_{\vec{p}}\mathcal{L}_{pen} = \vec{p}\mathbf{V}^{-1} + \mathbf{V}^{-1}\vec{p} = 2\vec{p}\mathbf{V}^{-1} \tag{7}$$

Where the final simplification can happen since the error matrix, $\mathbf{V}^{-1}$, is guaranteed to be real and symmetric.

The gradient of the statistical likelihood, $\partial_{\vec{p}}\mathcal{L}_{stat}$ the sum of gradients of the joint likelihoods for all bins

$$\partial_{\vec{p}}\mathcal{L}_{stat} = \sum_b \partial_{\vec{p}}\mathcal{L}_b \tag{8}$$

The gradient of the likelihood for each bin is then

$$\partial_{\vec{p}}\mathcal{L}_b = \partial_{\vec{p}}J(D_b, W_b) = \frac{\partial}{\partial W_b}J(D_b, W_b) \times \partial_{\vec{p}}W_b. \tag{9}$$

Since the joint likelihood, J, is a scalar, the partial of J with respect to W_b is a scalar, and it can be calculated as a result of a floating point C++ function. The gradient of the expectation in the bin must be calculated from the event weights. The implication of Eqn. 9 is that we will need to separately calculate the gradient of the expectation for each bin of the likelihood.[1]

[1]This is a place where there might be simplifications, but I'm not seeing them off the top of my head.

The expectation in bin b is W_b which is calculated as the sum of the event weights w_e for the events, e in the bin, b

$$W_b = \sum_{e \in (\text{bin } b)} w_e \tag{10}$$

and

$$\partial_{\vec{p}} W_b = \sum_{e \in (\text{bin } b)} \partial_{\vec{p}} w_e \tag{11}$$

The event weight, w_e, is the product of the event dials, $D_i(\vec{p}$ for the event

$$w_e = \prod_i D_i \tag{12}$$

where the dials are a function of the parameters, $\vec{p}$.

The gradient of each event weight is then

$$\partial_{\vec{p}} w_e = w_e \sum_i \frac{\partial_{\vec{p}} D_i}{D_i} \tag{13}$$

The values of w_e, D_i, and $\partial_{\vec{p}} D_i$ can be directly calculated in C++.[2]

3 Task to add the gradient to GUNDAM

- Define an API for the gradient in the direct calculation of the dials
- Define an API for the gradient in the CacheManager calculation of the event weights.
- Find a way to efficiently calculate and store the gradient of the expectation for each bin. Unfortunately (see Eqn. 9), the gradient of the expectation in each bin is required to calculate the gradient of the likelihood for each bin. The form needs to be "equivalent" for the CacheManager and direct calculation (i.e. have very similar layouts in memory) so that the final likelihood calculation is equivalent for bot.

[2] w_e is currently calculated as the eventWeight, and D_i is the dial value. The gradient of the dial, $\partial_{\vec{p}} D_i$ needs to be added to GUNDAM