0.1 Model: $\Xi^{ch}K^{ch}$

When fitting the $\Xi^-(\bar{\Xi}^+)\mathrm{K}^\pm$ results, it is necessary to include both strong and Coulomb effects. In this case, Equation **??** is no longer valid, and, in fact, there is no analytical form with which to fit. Therefore, we must begin with the wave function describing the pair interaction, and simulate many particle pairs to obtain a theoretical fit correlation function. The code developed to achieve this functionality is called "CoulombFitter". Currently, in order to generate the statistics needed for a stable fit, we find that $\sim 10^4$ simulated pairs per 10 MeV bin are necessary. Unfortunately, the nature of this process means that the "CoulombFitter" takes much longer to run than the "LednickyFitter" of Section **??**.

The two-particle correlation function may be written as:

$$C(\mathbf{k}^*) = \sum_S \rho_S \int S(\mathbf{r}^*)|\Psi^S_{\mathbf{k}^*}(\mathbf{r}^*)|^2 d^3\mathbf{r}^* \tag{1}$$

where ρ_S is the normalized emission probability of particles in a state with spin S, $S(\mathbf{r}^*)$ is the pair emission source distribution (assumed to be Gaussian), and $\Psi^S_{\mathbf{k}^*}(\mathbf{r}^*)$ is the two-particle wave-function including both strong and Coulomb interactions [**?**]:

$$\Psi_{\mathbf{k}^*}(\mathbf{r}^*) = e^{i\delta_c}\sqrt{A_c(\eta)}[e^{i\mathbf{k}^*\cdot\mathbf{r}^*}F(-i\eta,1,i\xi) + f_c(k^*)\frac{\tilde{G}(\rho,\eta)}{r^*}] \tag{2}$$

where $\rho = k^*r^*$, $\eta = (k^*a_c)^{-1}$, $\xi = \mathbf{k}^*\cdot\mathbf{r}^* + k^*r^* \equiv \rho(1+\cos\theta^*)$, and $a_c = (\mu z_1 z_2 e^2)^{-1}$ is the two-particle Bohr radius (including the sign of the interaction). δ_c is the Coulomb s-wave phase shift, $A_c(\eta)$ is the Coulomb penetration factor, $\tilde{G} = \sqrt{A_c}(G_0 + iF_0)$ is a combination of the regular (F_0) and singular (G_0) s-wave Coulomb functions. $f_c(k^*)$ is the s-wave scattering amplitude:

$$f_c(k^*) = [\frac{1}{f_0} + \frac{1}{2}d_0k^{*2} - \frac{2}{a_c}h(\eta) - ik^*A_c(\eta)]^{-1} \tag{3}$$

where, the "h-function", $h(\eta)$, is expressed through the digamma function, $\psi(z) = \Gamma'(z)/\Gamma(z)$ as:

$$h(\eta) = 0.5[\psi(i\eta) + \psi(-i\eta) - \ln(\eta^2)] \tag{4}$$

As stated before, to generate a fit correlation function, we must simulate a large number of pairs, calculate the wave-function, and average Ψ^2 over all pairs in a given k* bin. Essentially, we calculate Equation 1 by hand:

$$\begin{aligned} C(\mathbf{k}^*) &= \sum_S \rho_S \int S(\mathbf{r}^*)|\Psi^S_{\mathbf{k}^*}(\mathbf{r}^*)|^2 d^3\mathbf{r}^* \\ &\longrightarrow C(|\mathbf{k}^*|) \equiv C(k^*) = \sum_S \rho_S \langle|\Psi^S(\mathbf{k}^*_\mathbf{i},\mathbf{r}^*_\mathbf{i})|^2\rangle_i \\ &\longrightarrow C(k^*) = \lambda\sum_S \rho_S \langle|\Psi^S(\mathbf{k}^*_\mathbf{i},\mathbf{r}^*_\mathbf{i})|^2\rangle_i + (1-\lambda) \end{aligned} \tag{5}$$

where $\langle\rangle_i$ represents an average over all pairs in a given k* bin.

In summary, for a given k* bin, we must draw $N_{pairs} \sim 10^4$ pairs, and for each pair:

1. Draw a random $\mathbf{r}^*$ vector according to our Gaussian source distribution $S(\mathbf{r}^*)$

2. Draw a random $\mathbf{k}^*$ vector satisfying the $|\mathbf{k}^*|$ restriction of the bin
   - We draw from real k^* vectors obtained from the data
   - However, we find that drawing from a distribution flat in k^* gives similar results
3. Construct the wave-function Ψ

After all pairs for a given k* bin are simulated and wave-functions obtained, the results are averaged to give the fit result.

Construction of the wave-functions, Equation 2, involves a number of complex functions not included in standard C++ or ROOT libraries (namely, $h(\eta)$, $\tilde{G}(\rho,\eta)$), and $F(-i\eta,1,i\xi)$. These functions were even difficult to find and implement from elsewhere. Our solution was to embed a Mathematica kernel into our C++ code to evaluate these functions. However, having Mathematica work on-the-fly with the fitter was far too time consuming (fitter would have taken day, maybe weeks to finish). Our solution was to use Mathematica to create matrices representing these functions for different parameter values. During fitting, these matrices were then interpolated and the results used to build the wave-functions. This method decreased the running time dramatically, and we are not able to generate results in under $\sim$ 1 hour. This process will be explained in more detail in future versions of the note.