# XMBF 2.40

Stefan Meinel

June 7, 2013

## Contents

**1 Introduction** **3**

**2 Compiling XMBF** **3**
- 2.1 Standard Double-Precision Build . . . 3
- 2.2 Build with Quad-Double Inverter . . . 3
- 2.3 Build for parallel bootstrap with MPI . . . 4
- 2.4 Additional programs . . . 4

**3 Using XMBF** **4**
- 3.1 Using `XMBF_mpack_qd` . . . 5
- 3.2 Using `XMBF_mpi` . . . 5

**4 Basic structure of the input file** **5**

**5 The `macros` node** **6**

**6 The `combined_models` node** **6**

**7 The `chi_sqr_extra_term` node** **10**

**8 The `fit_settings` node** **11**

**9 The `parameter_values` and `constant_values` nodes** **12**

**10 More details on fit ranges** **14**
- 10.1 Lower and upper bounds . . . 14
- 10.2 Step sizes . . . 14

**11 More details on bootstrap** **14**
- 11.1 Resampling the data . . . 14
- 11.2 Resampling both the data and the fit ranges . . . 15

**12 Multifit** **16**

**13 Built-in Models** **16**
13.1 Scalar two-point models . . . . . 16
13.1.1 multi_exp_model . . . . . 16
13.1.2 multi_exp_expE_model . . . . . 17
13.1.3 multi_exp_Asqr_model . . . . . 17
13.1.4 multi_exp_Asqr_expE_model . . . . . 17
13.1.5 multi_alt_exp_model . . . . . 17
13.1.6 multi_alt_exp_expE_model . . . . . 18
13.1.7 multi_alt_exp_Asqr_model . . . . . 18
13.1.8 multi_alt_exp_Asqr_expE_model . . . . . 18
13.2 Vector two-point models . . . . . 19
13.3 Two-point models with periodic B.C. . . . . . 19
13.4 Two-point models with time-independent contributions . . . . . 20
13.5 Matrix two-point models . . . . . 22
13.5.1 multi_exp_mat_model . . . . . 22
13.5.2 multi_exp_expE_mat_model . . . . . 22
13.5.3 multi_alt_exp_mat_model . . . . . 23
13.5.4 multi_alt_exp_expE_mat_model . . . . . 23
13.6 Matrix two-point models, type II . . . . . 23
13.6.1 multi_exp_mat_II_model . . . . . 24
13.6.2 multi_exp_expE_mat_II_model . . . . . 24
13.7 Triangular matrix two-point models . . . . . 24
13.7.1 multi_exp_mat_upper_model . . . . . 24
13.7.2 multi_exp_expE_mat_upper_model . . . . . 25
13.7.3 multi_exp_mat_II_upper_model . . . . . 25
13.7.4 multi_exp_expE_mat_II_upper_model . . . . . 25
13.8 Non-symmetric matrix two-point models . . . . . 26
13.8.1 multi_exp_nonsym_mat_model . . . . . 26
13.8.2 multi_exp_expE_nonsym_mat_model . . . . . 26
13.8.3 multi_alt_exp_nonsym_mat_model . . . . . 27
13.8.4 multi_alt_exp_expE_nonsym_mat_model . . . . . 28
13.9 Scalar three-point models . . . . . 28
13.9.1 threept_multi_exp_model . . . . . 28
13.9.2 threept_multi_exp_expE_model . . . . . 29
13.9.3 threept_multi_alt_exp_model . . . . . 29
13.9.4 threept_multi_alt_exp_expE_model . . . . . 31
13.10 Vector three-point models . . . . . 31
13.11 "Degenerate" three-point models . . . . . 32
13.11.1 threept_constant_model . . . . . 32
13.11.2 threept_constant_sqr_model . . . . . 32
13.11.3 multi_exp_2var_model . . . . . 32
13.11.4 multi_exp_expE_2var_model . . . . . 33
13.12 Multi-particle two-point models . . . . . 33
13.12.1 multi_part_exp_expE_model . . . . . 33

**14 User-defined model** **34**

# 1 Introduction

XMBF is used to perform fits that can combine multiple QMBF-type fit models, each with its own data file, to a simultaneous, fully correlated fit. See the documentation of QMBF for the meaning of “QMBF-type fit model” and the data file format. More information on the method of combining multiple models to a simultaneous correlated fit can be found in Appendix C.2 of the PhD thesis “Heavy quark physics on the lattice with improved nonrelativistic actions”, available at http://www.dspace.cam.ac.uk/handle/1810/225126.

In XMBF, fit parameters with the same name across different models will be shared. The parameter names for each model are defined using input from the user, so that the user can decide which parameters will be common to which fit models. XMBF uses XML input files to specify the models, fitting ranges etc. The XML files are read such that for each node, the *order* of the child nodes does not matter. Comments are also allowed, using the standard XML syntax for comments.

To allow the correct calculation of correlations, the data files for each model must correspond to the same order of measurements.

# 2 Compiling XMBF

## 2.1 Standard Double-Precision Build

Required libraries are

- GNU Scientific Library, version $\geq 1.13$
- libxml++
- Boost C++ libraries

and their dependencies. When installing these libraries using a package manager, note that the *development* packages are also needed (these usually have `-dev` or `-devel` in the package name). A Makefile is supplied with the source code; the variables `INCPATH` and `LIBS` may require adjustment for the specific machine.

## 2.2 Build with Quad-Double Inverter

It is possible to compile a version of XMBF that uses the libraries

- MPACK Multiple precision arithmetic BLAS (MBLAS) and LAPACK (MLAPACK),
- QD Quad Double package,

to invert the data correlation matrix in “quad double” precision (256 bits, approx. 64 digits). This is useful to prevent round-off errors when the data correlation matrix has a condition number larger than $10^{16}$. Note that most numerical operations in the quad-double build of XMBF are still performed in standard double precision; only the inversion (or pseudo-inversion) of the data correlation matrix (and some related operations) are performed in quad double precision.

To compile this higher-precision version, install MPACK and QD, and then compile XMBF using the makefile `Makefile_mpack_qd` (after adjusting the variables `INCPATH` and `LIBS`). This generates an executable called `XMBF_mpack_qd`.

When doing a fit with `XMBF_mpack_qd`, if the XML input file specifies the inversion method `LU` (see Sec. 12), then MPACK’s functions `Rgetrf` and `Rgetri` are used to fully invert the data correlation matrix. If the XML input file instead specifies `svd_fixed_cut`, `svd_ratio_cut`, or `svd_absolute_-cut` (see Sec. 12), the MPACK function `Rsyev` is used to compute the spectral decomposition of the data correlation matrix, and then the pseudo-inverse, removing the contributions from the smallest eigenvalues as determined by the user. Note that with version 0.6.7 of MPACK, the function `Rsyev` may fail for large matrices (dimension more than about 500) for an unknown reason, in which case XMBF will abort.

### 2.3 Build for parallel bootstrap with MPI

It is possible to build a version of XMBF that performs bootstrap (see Sec. 11) in parallel, using MPI. To compile this version, use `make -f Makefile_mpi` (after adjusting this Makefile for your machine, if necessary). This generates an MPI executable called `XMBF_mpi`. Note that this executable has restricted functionality (bootstrap only).

### 2.4 Additional programs

The source of XMBF includes some additional programs for manipulating the XML files. These programs can be found in subdirectories and must be compiled separately. The most important tool is called `MBF_to_XMBF`, and allows to convert .mbf session files generated by QMBF into XML input files for use with XMBF. The usage is as follows:

```
MBF_to_XMBF mbf_file xml_file
```

## 3 Using XMBF

XMBF requires an XML input file containing all the settings, e.g. the fit models, start values for the parameters, the locations of the data files. The usage is as follows:

```
XMBF [options] inputfile

options:
  -o output.xml        write fit results to "output.xml"
  -c cov.dat           write covariance matrix to "cov.dat"
  -r res.dat           write results to "res.dat"
  -re res_err.dat      write results with errors to "res_err.dat"
  -p directory         plot data and fit functions, write output files to "directory"
  -b directory         perform bootstrap, write output files to "directory"
  -m directory         perform multifit, write output files to "directory"
  -v level             verbose level [0,1,2]
```

XMBF always reads the input file and performs a fit; the results are printed to stdout. When the option `-o outputfile` is given, the fit results are additionally written to an output file in XML format. When the options `-c cov.dat` is specified, the elements of the parameter covariance matrix are written to the specified file (similarly, `-r res.dat` writes the central values of the fitted parameters; `-re res_err.dat` writes the central values and errors). With the option `-p directory`, the averaged data and the values of the fitted functions are written to files in the specified `directory`; these files can then be used for plotting. With the option `-b directory`, XMBF performs the bootstrap procedure (see Sec. 11) and writes the results for each parameter into an individual file in `directory`. With the option `-m directory`, XMBF performs the "multifit" procedure (see Sec. 12) and writes the results for each parameter into an individual file in `directory`.

The option `-v level` determines how much information is printed to stdout during the fit. The default corresponds to `-v 0`. With `-v 1`, after every iteration the current values of $\chi^2$/dof and $\lambda$ are printed; with `-v 2` in addition the current values of all parameters are printed after every step.

### 3.1 Using XMBF_mpack_qd

The quad-double version, `XMBF_mpack_qd`, can be used in the same way as the standard version.

### 3.2 Using XMBF_mpi

The MPI version, `XMBF_mpi`, is only intended for parallel bootstrap (see Sec. 11). It must be used as follows (assuming Open MPI):

```
mpirun -np nprocs XMBF_mpi -b directory inputfile
```

Here, `nprocs` is the number of MPI processes to be used; the number of bootstrap samples must be divisible by `nprocs`.

## 4 Basic structure of the input file

This is the basic structure of an input file:

```
<?xml version="1.0"?>

<fit>

  <macros>
    ...
  </macros>

  <combined_model>
    ...
  </combined_model>

  <chi_sqr_extra_term>
  ...
  </chi_sqr_extra_term>

  <fit_settings>
    ...
  </fit_settings>

  <parameter_values>
    ...
  </parameter_values>

  <constant_values>
    ...
  </constant_values>

</fit>
```

There has to be a root node, called `fit`, which contains (up to) five main nodes (in arbitrary order):

- `macros` [optional]: cf. section 5

- `combined_model` : cf. section 6
- `chi_sqr_extra_term` [optional] : cf. section 7
- `fit_settings` : cf. section 12
- `parameter_values` : cf. section 9
- `constant_values` [optional]: cf. section 9

## 5 The `macros` node

Here is an example:

```
<macros>

  <macro>
    <name> INITIAL_dE_START_VAL </name>
    <value> -1.1 </value>
  </macro>
  <macro>
    <name> INITIAL_dE_PRIOR </name>
    <value> -1 </value>
  </macro>
  <macro>
    <name> INITIAL_dE_PRIOR_WIDTH </name>
    <value> 1 </value>
  </macro>
  ...
</macros>
```

The `macros` node contains a list of macros, each of them with a unique name and a value. Both are strings; spaces and newlines will be removed. For the example shown here, appearences of `INITIAL_dE_START_VAL` in content nodes elsewhere in the XML document will be replaced by `-1.1` etc.

## 6 The `combined_models` node

The `combined_models` node contains one or more individual models, to be fitted simultaneously. Each individual model can either be a built-in model (cf. Sec. 13) or a completely user-defined model (cf. Sec. 14).

An example for a `combined_models` node with 5 built-in models is shown below:

```
<combined_model>

  <multi_alt_exp_expE_mat_model>  <!-- B 2-point fn -->
    <n_exp> 8< /n_exp>
    <n_o_exp> 8 </n_o_exp>
    <A_name> Ai </A_name>
    <B_name> Bi </B_name>
    <E_name> Ei </E_name>
    <dE_name> dEi </dE_name>
    <t_name> t </t_name>
    <dim_1> 2 </dim_1>
    <dim_2> 1 </dim_2>
```

```
    <fit_domain>
      <variable_name> t </variable_name>
      <range>
        <min> 2 </min>
        <max> 32 </max>
      </range>
    </fit_domain>
    <plot_domain>
      <variable_name> t </variable_name>
      <plot_order> 1 </plot_order>
      <range>
        <min> 0 </min>
        <max> 32 </max>
      </range>
      <step> 1 </step>
    </plot_domain>
    <data_file>
      <file_type> ASCII </file_type>
      <file_name> re_B_2x1_matrix_mom_0_0_0.dat </file_name>
    </data_file>
  </multi_alt_exp_expE_mat_model>

  <multi_alt_exp_Asqr_expE_BC_model>  <!-- Kstar 2-point fn -->
    <n_exp> 8 </n_exp>
    <n_o_exp> 8 </n_o_exp>
    <A_name> Af </A_name>
    <B_name> Bf </B_name>
    <E_name> Ef </E_name>
    <dE_name> dEf </dE_name>
    <t_name> t </t_name>
    <T_name> L4 </T_name>
    <fit_domain>
      <variable_name>t</variable_name>
      <range>
        <min> 1 </min>
        <max> 20 </max>
      </range>
      <range>
        <min> 44 </min>
        <max> 63 </max>
      </range>
    </fit_domain>
    <plot_domain>
      <variable_name> t </variable_name>
      <plot_order> 1 </plot_order>
      <range>
        <min> 0 </min>
        <max> 64 </max>
      </range>
      <step> 0.01 </step>
    </plot_domain>
    <data_file>
      <file_type> ASCII </file_type>
      <file_name> re_v_ls_xyz_mom_0_0_0.dat </file_name>
    </data_file>
  </multi_alt_exp_Asqr_expE_BC_model>

  <threept_multi_alt_exp_expE_model>   <!-- B to Kstar 3-point fn -->
```

```
    <n_exp_initial> 8 </n_exp_initial>
    <n_o_exp_initial> 8 </n_o_exp_initial>
    <n_exp_final> 8 </n_exp_final>
    <n_o_exp_final> 8 </n_o_exp_final>
    <A_name> A </A_name>
    <B_name> B </B_name>
    <E_initial_name> Ei </E_initial_name>
    <dE_initial_name> dEi </dE_initial_name>
    <E_final_name> Ef </E_final_name>
    <dE_final_name> dEf </dE_final_name>
    <t_name> t </t_name>
    <T_name> T </T_name>
    <fit_domain>
      <variable_name> t </variable_name>
      <range>
        <min> 1 </min>
        <max> T-2 </max>
      </range>
    </fit_domain>
    <fit_domain>
      <variable_name> T </variable_name>
      <range>
        <min> 14 </min>
        <max> 16 </max>
      </range>
    </fit_domain>
    <plot_domain>
      <variable_name> t </variable_name>
      <plot_order> 2 </plot_order>
      <range>
        <min> 1 </min>
        <max> T-2 </max>
      </range>
      <step> 1 </step>
    </plot_domain>
    <plot_domain>
      <variable_name> T </variable_name>
      <plot_order> 1 </plot_order>
      <range>
        <min> 14 </min>
        <max> 16 </max>
      </range>
      <step> 1 </step>
    </plot_domain>
    <data_file>
      <file_type>binary</file_type>
      <file_name> im_gj_s0jg5_sl_pf_0_0_0_p_0_0_0.bin </file_name>
    </data_file>
  </threept_multi_alt_exp_expE_model>

  <multi_exp_Asqr_expE_BC_model>   <!-- K 2-point fn -->
    <n_exp> 1 </n_exp>
    <A_name> KAf </A_name>
    <B_name> KBf </B_name>
    <E_name> KEf </E_name>
    <dE_name> KdEf </dE_name>
    <t_name> t </t_name>
    <T_name> L4 </T_name>
```

```
    <fit_domain>
      <variable_name> t </variable_name>
      <range>
        <min> 10 </min>
        <max> 54 </max>
      </range>
    </fit_domain>
    <plot_domain>
      <variable_name> t </variable_name>
      <plot_order> 1 </plot_order>
      <range>
        <min> 0 </min>
        <max> 64 </max>
      </range>
      <step> 0.01 </step>
    </plot_domain>
    <data_file>
      <file_type> ASCII </file_type>
      <file_name> re_ps_ls_mom_0_0_0.dat </file_name>
    </data_file>
  </multi_exp_Asqr_expE_BC_model>

  <threept_multi_alt_exp_expE_model>      <!-- B to K 3-point fn -->
    <n_exp_initial> 1 </n_exp_initial>
    <n_o_exp_initial> 1 </n_o_exp_initial>
    <n_exp_final> 1 </n_exp_final>
    <n_o_exp_final> 0 </n_o_exp_final>
    <A_name> KA </A_name>
    <B_name> KB </B_name>
    <E_initial_name> Ei </E_initial_name>
    <dE_initial_name> dEi </dE_initial_name>
    <E_final_name> KEf </E_final_name>
    <dE_final_name> KdEf </dE_final_name>
    <t_name> t </t_name>
    <T_name> T </T_name>
    <fit_domain>
      <variable_name> t </variable_name>
      <range>
        <min> 6 </min>
        <max> T-12 </max>
      </range>
    </fit_domain>
    <fit_domain>
      <variable_name> T </variable_name>
      <range>
        <min> 0 </min>
        <max> 26 </max>
      </range>
    </fit_domain>
    <plot_domain>
      <variable_name> t </variable_name>
      <plot_order> 2 </plot_order>
      <range>
        <min> 0 </min>
        <max> T </max>
      </range>
      <step> 1 </step>
    </plot_domain>
```

```
      <plot_domain>
        <variable_name> T </variable_name>
        <plot_order> 1 </plot_order>
        <range>
          <min> 0 </min>
          <max> 26 </max>
        </range>
        <step> 1 </step>
      </plot_domain>
      <data_file>
        <file_type> binary </file_type>
        <file_name> re_g5_g0_sl_pf_0_0_0_p_0_0_0.bin </file_name>
      </data_file>
    </threept_multi_alt_exp_expE_model>

</combined_model>
```

Every model has the nodes fit_domain (see Sec. 10 for more details), data_file, and plot_domain (the latter is only needed when plotting). The allowed values for the property file_type are ASCII and binary. For the other properties, see the descriptions of the individual models in Secs. 13 and 14. For the built-in models, he strings entered in fields such as dE_name are used as templates for the parameter names. XMBF will "decorate" the names with additional indices as appropriate; see Sec. 13 for more details.

Fit parameters (and constants) with the same names will be shared between individual models, i.e. they are forced to have the same value globally. Note that *variable names* are only used individually for each model and only serve to specify the individual fitting ranges. It does not matter whether two models have a common variable name or not.

## 7 The chi_sqr_extra_term node

The optional chi_sqr_extra_term node can be used to add an arbitrary function of the fit parameters to $\chi^2$. Note that it must be enabled separately by putting the line "<chi_sqr_extra_term_enabled> true </chi_sqr_extra_term_enabled>" in the fit_settings node. Here is an example, which forces two fit parameters a and b to have similar values within a width sigma_a_b, and puts a non-gaussian constraint on another parameter c:

```
  <chi_sqr_extra_term>
   <function> sqr(a-b)/sqr(sigma_a_b) + exp(sqr(c-1.0)/sqr(sigma_c)) </function>
   <constant>
     <name> sigma_a_b </name>
     <value> 0.1 </value>
   </constant>
   <constant>
     <name> sigma_c </name>
     <value> 0.01 </value>
   </constant>
   <num_diff_step> 1e-8 </num_diff_step>
 </chi_sqr_extra_term>
```

In general, the function can be constructed using the elementary operations

```
+, -, *, /,
(, ),
exp(...), log(...),
sin(...), cos(...), tan(...),
sinh(...), cosh(...), tanh(...),
arcsin(...), arccos(...), arctan(...),
sqr(...), sqrt(...),
alt(...).
```

The function may contain any of the fit parameters resulting from the models in `combined_model`. It may also contain the constants defined inside `chi_sqr_extra_term`, as well as numerical literal values. The derivatives of the function with respect to the fit parameters are calculated numerically using the step size defined via `num_diff_step`.

## 8 The `fit_settings` node

```
<fit_settings>
  <restrict_data_range> false </restrict_data_range>
  <data_range_min> 1 </data_range_min>
  <data_range_max> 1000 </data_range_max>
  <chi_sqr_extra_term_enabled> false </chi_sqr_extra_term_enabled>
  <bayesian> true </bayesian>
  <random_priors> true </random_priors>
  <num_diff_first_order> false </num_diff_first_order>
  <second_deriv_covariance> false </second_deriv_covariance>
  <second_deriv_minimization> false </second_deriv_minimization>
  <num_diff_step> 1e-08 </num_diff_step>
  <start_lambda> 0.001 </start_lambda>
  <lambda_factor> 10 </lambda_factor>
  <chi_sqr_tolerance> 0.001 </chi_sqr_tolerance>
  <chi_sqr_per_dof_tolerance> true </chi_sqr_per_dof_tolerance>
  <n_parameters_dof> 18 </n_parameters_dof>
  <inversion_method> LU </inversion_method>
  <bootstrap_normalization> false </bootstrap_normalization>
  <svd_fixed_cut> 0 </svd_fixed_cut>
  <svd_ratio_cut> 1e-06 </svd_ratio_cut>
  <svd_absolute_cut> 1e-06 </svd_absolute_cut>
  <max_iterations> 1000 </max_iterations>
  <bin_size> 1 </bin_size>
  <bootstrap_samples> 500 </bootstrap_samples>
  <use_bse_file> true </use_bse_file>
  <bse_file> bootstrap_1600configs_500samples.bse </bse_file>
  <restrict_bootstrap_range>false</restrict_bootstrap_range>
  <bootstrap_range_min>1</bootstrap_range_min>
  <bootstrap_range_max>50</bootstrap_range_max>
</fit_settings>
```

Most of the settings are as in to QMBF; see the documentation of QMBF.

The property `chi_sqr_per_dof_tolerance` specifies whether the numerical value entered in `chi_sqr_tolerance` (the abortion criterion for the Levenberg-Marquardt iteration) is used for the total $\chi^2$ or for $\chi^2/\mathrm{dof}$.

The optional property n_parameters_dof is analoguous to QMBF's setting "Number of parameters to be subtracted from d.o.f". If this property is not present in the XML file, the default values are: 0 (for Bayesian fits), or the total number of parameters in the fit (for non-Bayesian fits).

The property inversion_method specifies the method used in the calculation of the inverse of the data correlation matrix. Allowed values are:

- LU (for full inversion)
- svd_fixed_cut (remove given number of smallest eigenvalues)
- svd_ratio_cut (remove eigenvalues that are smaller than some given fraction of the largest eigenvalue)
- svd_absolute_cut (remove eigenvalues smaller than some given value)
- diagonal (keep only diagonal elements in data correlation matrix)

One very important setting is bootstrap_normalization. If set to false, the data correlation matrix will be normalized with the usual factor of

$$\frac{1}{N(N-1)},$$

where $N$ is the number of data sets. If set to true, the data correlation matrix will instead be normalized with the factor

$$\frac{1}{N-1}.$$

This is needed to get the correct error estimates for fit parameters for the case that the original data file was created using bootstrap over data sets (e.g. in the calculation of ratios of three-point and two-point functions).

For more details on the setting concerning bootstrap, see Sec. 11.

## 9 The parameter_values and constant_values nodes

An excerpt of the parameter_values node from an example XML file is shown below. Note: entries with parameter names that are not needed for the models are allowed and will be simply be ignored. This is very convenient if for example changing the number of exponentials in a fit. Also note: the tags prior and prior_width for each parameter are only needed for Bayesian fits, which are enabled using <bayesian> true </bayesian> in the fit_settings node (see Sec. 12).

```
<parameter_values>

  <parameter>
    <name> Ei </name>
    <start_value> -0.66 </start_value>
    <prior> -0.6622 </prior>
    <prior_width> 0.04 </prior_width>
  </parameter>
  <parameter>
    <name>Eio</name>
    <start_value> -0.33 </start_value>
```

```
      <prior> -0.3270 </prior>
      <prior_width> 0.3 </prior_width>
    </parameter>
    <parameter>
      <name>Ai__1</name>
      <start_value> 0.054 </start_value>
      <prior> 0.0544017 </prior>
      <prior_width> 0.007 </prior_width>
    </parameter>
    <parameter>
      <name>Aio__1</name>
      <start_value> 0.044 </start_value>
      <prior> 0.04416 </prior>
      <prior_width> 0.07 </prior_width>
    </parameter>
    <parameter>
      <name>Ai__2</name>
      <start_value> 0.18 </start_value>
      <prior> 0.179835 </prior>
      <prior_width> 0.03 </prior_width>
    </parameter>
    <parameter>
      <name>Aio__2</name>
      <start_value> 0.16 </start_value>
      <prior> 0.16123 </prior>
      <prior_width> 0.2 </prior_width>
    </parameter>
    ...

    <parameter>
      <name> dEi_1 </name>
      <start_value> INITIAL_dE_START_VAL </start_value>
      <prior> INITIAL_dE_PRIOR </prior>
      <prior_width> INITIAL_dE_PRIOR_WIDTH </prior_width>
    </parameter>
    <parameter>
      <name> dEi_2 </name>
      <start_value> INITIAL_dE_START_VAL </start_value>
      <prior> INITIAL_dE_PRIOR </prior>
      <prior_width> INITIAL_dE_PRIOR_WIDTH </prior_width>
    </parameter>
    <parameter>
      <name> dEi_3 </name>
      <start_value> INITIAL_dE_START_VAL </start_value>
      <prior> INITIAL_dE_PRIOR </prior>
      <prior_width> INITIAL_dE_PRIOR_WIDTH </prior_width>
    </parameter>
    ...

  </parameter_values>
```

Note that in the above example, macros named `INITIAL_dE_START_VAL` etc. are used, as defined in the example shown in Sec. 5.

Shown below is an example for the `constant_values` node:

```
  <constant_values>

    <constant>
```

```
      <name> L4 </name>
      <value> 64 </value>
    </constant>

  </constant_values>
```

# 10 More details on fit ranges

Every model inside `combined_models` (see Sec. 6) must have one node called `fit_domain` for each variable of that model.

Every `fit_domain` node can contain an arbitrary number of `range` nodes, where every range node specifies a condition on the value of the variable (required: `min`, `max` – see Sec. 10.1; optional: `step` – see Sec. 10.2). Every fit domain is the *union* (rather than the intersection) of the individual `range`s. That is, a fit point is included in the fit if and only if the value of every variable at that point satisfies the condition of *at least one* of its range nodes.

## 10.1 Lower and upper bounds

Every `range` node must have two entries called `min` and `max`. These entries can contain numerical values (integer or floating point), but also functions of other variables and constants (for example, in the model `threept_multi_alt_exp_expE_model` shown in Sec. 6, the range for the variable `t` is `<min>1</min>` and `<max>T-2</max>`, where `T` is the other variable. The function strings are parsed by XMBF; the allowed operations are the same as given in Sec. 14. If no `step` (see Sec. 10.2) is specified in the `range` node, a value $x$ of a variable is considered inside the range if $\texttt{min} \leq x \leq \texttt{max}$.

## 10.2 Step sizes

Every `range` node can contain an optional entry called `step`, which must contain a step size $\Delta$, which is a positive numerical value (integer or floating point). In this case, a value $x$ of a variable is considered inside the range if in addition to statisfying $\texttt{min} \leq x \leq \texttt{max}$, the value satisfies as $x = \texttt{min} + n \cdot \Delta$, where $n$ is an integer.

# 11 More details on bootstrap

## 11.1 Resampling the data

When starting XMBF with the command-line option `-b directory`, the program performs the bootstrap procedure and writes the results for each parameter into an individual file in `directory`. The output files have names that are combinations of the XML input file name and the parameter names. On multi-core systems, the bootstrap procedure can be parallelized using `XMBF_mpi` instead of `XMBF` (see Sec. 3.2).

The number of bootstrap samples is specified in the `fit_settings` node via `bootstrap_samples`. Each bootstrap sample is obtained by randomly choosing $N$ out of the $N$ data configurations with allowed repetitions. XMBF will recompute and invert the data correlation matrix for every single bootstrap sample.

When the bootstrap is completed, the bootstrap averages and error estimates (based on sorting the results and taking the 68% central part of the distributions) of the fit results are also printed to the standard output.

By default, the random numbers of configurations are generated by XMBF at run time just before the bootstrap. Alternatively, if the option `use_bse_file` in the `fit_settings` node is set to `true`, the numbers are read from a text file, specified in `bse_file`. The format of a bootstrap ensemble file is as follows: the number of bootstrap samples $S$, followed by the number of configurations $N$, followed by $S \cdot N$ random integer numbers in the range $1...N$. Such files can also be generated by QMBF (see the QMBF manual).

For Bayesian fitting, it is recommended to set the option `random_priors` to `true`, in order to get the (approximately) correct probability distribution. If this is activated, in addition to randomly choosing data set ensembles, the priors will be choosen randomly from Gaussian distributions with the given prior widths. This option does not affect the bootstrap for non-Bayesian fits.

Finally, by setting the optional property `restrict_bootstrap_range` to `true`, the bootstrap procedure can be restricted to a certain range of the samples, specified using `bootstrap_range_min` and `bootstrap_range_max`. When a restricted range in the input XML file is specified, only only the appropriate part of the .bse file will be used. Note that the option `restrict_bootstrap_range` is fully compatible with `random_priors`: if `bootstrap_range_min` is greater than 1, the correct amount of random numbers is skipped, so that also the sequence of random numbers for the priors remains unchanged.

### 11.2 Resampling both the data and the fit ranges

XMBF can performed a generalized bootstrap procedure such that at the same time as resampling the data configurations (and parameter priors, if activated), the *fit ranges* for the variables are also resampled. This allows the incorporation of systematic errors due to the choices of the fitting ranges into the bootstrap distribution.

Every `range` node can have an optional entry called `range_bootstrap_file`, which contains the name of a file with the ranges for bootstrap (generated by the user). For example:

```
<fit_domain>
  <variable_name> t </variable_name>
  <range>
    <min> 8 </min>
    <max> T-12 </max>
    <range_bootstrap_file> BK3pt.brf </range_bootstrap_file>
  </range>
</fit_domain>
<fit_domain>
  <variable_name>T</variable_name>
  <range>
    <min> 0 </min>
    <max> 26 </max>
  </range>
</fit_domain>
```

In this case, the explicit range specified for the variable `t` via `min` and `max` (8 to $T-12$) are only used for the initial fit when XMBF starts, but not for the bootstrap. During the bootstrap, the values for `min` and `max` for each bootstrap sample are instead read from the file specified in `range_bootstrap_file`, in this case "BK3pt.brf". This number of lines in this file must be equal to `bootstrap_samples`, and every line must have two entries, the min and max. As discussed in Sec. 10.1, these entries may also contain functions of constants and other variables (but the functions must not contain whitespaces). The first few lines of the file "BK3pt.brf" in the example could for example be

```
5    T-12
10   T-14
7    T-12
13   T-13
```

As discussed in Sec. 10.1, in XMBF every `fit_domain` can have multiple `range` nodes, and the actual fit range is the union of the ranges. It is allowed that some range nodes do not have a `range_bootstrap_file` while others do.

An optional `step` value (Sec. 10.2) inside a `range` node is respected during the bootstrap even if a `range_bootstrap_file` is used.

# 12 Multifit

When starting XMBF with the command-line option `-m directory`, the program performs the "multifit" procedure and writes the results for each parameter into an individual file in `directory`. The output files have names that are combinations of the XML input file name and the parameter names.

The "multifit" procedure is the following: XMBF performs $N$ successive fits, where the $n$th fit uses just the $n$th data sample instead of the average over all data samples. The covariance matrix stays fixed and is computed as usual using all data samples.

The "multifit" procedure is useful when the data samples themselves were obtained through a bootstrap procedure, and a corresponding resampling of the fit results is wanted.

For Bayesian fitting, it is recommended to set the option `random_priors` in the `fit_settings` node (see Sec. ) to `true`, in order to get the (approximately) correct probability distribution for the fit parameters. If this is activated, the priors for each fit will be drawn randomly from Gaussian distributions with the given prior widths and central values. This option does not affect the multifit procedure for non-Bayesian fits.

# 13 Built-in Models

In the following, those parts of the parameter, constant, and variable names that are specified by the user are typeset in `typewriter font`.

## 13.1 Scalar two-point models

### 13.1.1 `multi_exp_model`

- function(s):

$$f(\mathtt{t}) = \mathtt{A}\left[e^{-\mathtt{E}\ \mathtt{t}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\, e^{-(\mathtt{E}+\mathtt{dE\_}1+\ldots+\mathtt{dE\_}n)\mathtt{t}}\right]$$

- variable(s): `t`
- parameter(s): `A`, {`B_`$n$}, `E`, {`dE_`$n$}
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |

#### 13.1.2 multi_exp_expE_model

- function(s):

$$f(\mathtt{t}) = \mathtt{A}\left[e^{-e^{\mathtt{E}}\ \mathtt{t}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\, e^{-(e^{\mathtt{E}}+e^{\mathtt{dE\_1}}+\ldots+e^{\mathtt{dE\_}n})\mathtt{t}}\right]$$

- variable(s), parameter(s), properties: same as multi_exp_model

#### 13.1.3 multi_exp_Asqr_model

- function(s):

$$f(\mathtt{t}) = \mathtt{A}^2\left[e^{-\mathtt{E}\ \mathtt{t}} + \sum_{n=1}^{N-1} (\mathtt{B\_}n)^2\, e^{-(\mathtt{E}+\mathtt{dE\_1}+\ldots+\mathtt{dE\_}n)\mathtt{t}}\right]$$

- variable(s), parameter(s), properties: same as multi_exp_model

#### 13.1.4 multi_exp_Asqr_expE_model

- function(s):

$$f(\mathtt{t}) = \mathtt{A}^2\left[e^{-e^{\mathtt{E}}\ \mathtt{t}} + \sum_{n=1}^{N-1} (\mathtt{B\_}n)^2\, e^{-(e^{\mathtt{E}}+e^{\mathtt{dE\_1}}+\ldots+e^{\mathtt{dE\_}n})\mathtt{t}}\right]$$

- variable(s), parameter(s), properties: same as multi_exp_model

#### 13.1.5 multi_alt_exp_model

- function(s):

$$\begin{aligned} f(\mathtt{t}) \quad = \quad & \mathtt{A}\left[e^{-\mathtt{E}\ \mathtt{t}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\, e^{-(\mathtt{E}+\mathtt{dE\_1}+\ldots+\mathtt{dE\_}n)\mathtt{t}}\right] \\ & +(-1)^{\mathtt{t}+1}\mathtt{Ao}\left[e^{-\mathtt{Eo}\ \mathtt{t}} + \sum_{m=1}^{M-1} \mathtt{Bo\_}m\, e^{-(\mathtt{Eo}+\mathtt{dEo\_1}+\ldots+\mathtt{dEo\_}m)\mathtt{t}}\right] \end{aligned}$$

- variable(s): t

- parameter(s): $\texttt{A}$, $\{\texttt{B\_}n\}$, $\texttt{E}$, $\{\texttt{dE\_}n\}$, $\texttt{Ao}$, $\{\texttt{Bo\_}m\}$, $\texttt{Eo}$, $\{\texttt{dEo\_}m\}$
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| n_o_exp | $M$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |

### 13.1.6 multi_alt_exp_expE_model

- function(s):

$$\begin{aligned} f(\texttt{t}) &= \texttt{A}\left[e^{-e^{\texttt{E}}\,\texttt{t}} + \sum_{n=1}^{N-1} \texttt{B\_}n\, e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+\ldots+e^{\texttt{dE\_}n})\texttt{t}}\right] \\ &\quad +(-1)^{\texttt{t}+1}\texttt{Ao}\left[e^{-e^{\texttt{Eo}}\,\texttt{t}} + \sum_{m=1}^{M-1} \texttt{Bo\_}m\, e^{-(e^{\texttt{Eo}}+e^{\texttt{dEo\_1}}+\ldots+e^{\texttt{dEo\_}m})\texttt{t}}\right] \end{aligned}$$

- variable(s), parameter(s), properties: same as multi_alt_exp_model

### 13.1.7 multi_alt_exp_Asqr_model

- function(s):

$$\begin{aligned} f(\texttt{t}) &= \texttt{A}^2\left[e^{-\texttt{E}\,\texttt{t}} + \sum_{n=1}^{N-1} (\texttt{B\_}n)^2\, e^{-(\texttt{E}+\texttt{dE\_1}+\ldots+\texttt{dE\_}n)\texttt{t}}\right] \\ &\quad +(-1)^{\texttt{t}+1}\texttt{Ao}^2\left[e^{-\texttt{Eo}\,\texttt{t}} + \sum_{m=1}^{M-1} (\texttt{Bo\_}m)^2\, e^{-(\texttt{Eo}+\texttt{dEo\_1}+\ldots+\texttt{dEo\_}m)\texttt{t}}\right] \end{aligned}$$

- variable(s), parameter(s), properties: same as multi_alt_exp_model

### 13.1.8 multi_alt_exp_Asqr_expE_model

- function(s):

$$\begin{aligned} f(\texttt{t}) &= \texttt{A}^2\left[e^{-e^{\texttt{E}}\,\texttt{t}} + \sum_{n=1}^{N-1} (\texttt{B\_}n)^2\, e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+\ldots+e^{\texttt{dE\_}n})\texttt{t}}\right] \\ &\quad +(-1)^{\texttt{t}+1}\texttt{Ao}^2\left[e^{-e^{\texttt{Eo}}\,\texttt{t}} + \sum_{m=1}^{M-1} (\texttt{Bo\_}m)^2\, e^{-(e^{\texttt{Eo}}+e^{\texttt{dEo\_1}}+\ldots+e^{\texttt{dEo\_}m})\texttt{t}}\right] \end{aligned}$$

- variable(s), parameter(s), properties: same as multi_alt_exp_model

### 13.2 Vector two-point models

All the "scalar" two-point models listed in section 13.1 are also available as "vector" two-point models, called

- `multi_exp_vec_model`
- `multi_exp_expE_vec_model`
- `multi_exp_Asqr_vec_model`
- `multi_exp_Asqr_expE_vec_model`
- `multi_alt_exp_vec_model`
- `multi_alt_exp_expE_vec_model`
- `multi_alt_exp_Asqr_vec_model`
- `multi_alt_exp_Asqr_expE_vec_model`

These models require one further key in addition to the keys of the corresponding scalar models: the dimension of the vector:

| Key | content | type |
|---|---|---|
| `dim` | $i = 1...\mathtt{dim}$ | integer $\geq 1$ |

A vector model has then `dim` functions $f_i(\mathtt{t})$ ($i = 1...\mathtt{dim}$) of the same form as the underlying scalar model. These functions have individual amplitude parameters, but share all the energy parameters. For example, the functions for `multi_exp_vec_model` are

$$f_i(\mathtt{t}) = \mathtt{A\_\_}i \left[ e^{-\mathtt{E\ t}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\mathtt{\_\_}i \, e^{-(\mathtt{E}+\mathtt{dE\_1}+...+\mathtt{dE\_}n)\mathtt{t}} \right]$$

for $i = 1...\mathtt{dim}$.

### 13.3 Two-point models with periodic B.C.

All the "scalar" and "vector" two-point models listed in 13.1 and 13.2 are also available with periodic boundary conditions. The models with periodic boundary conditions are called

- `multi_exp_BC_model`
- `multi_exp_expE_BC_model`
- `multi_exp_Asqr_BC_model`
- `multi_exp_Asqr_expE_BC_model`
- `multi_alt_exp_BC_model`
- `multi_alt_exp_expE_BC_model`
- `multi_alt_exp_Asqr_BC_model`

- multi_alt_exp_Asqr_expE_BC_model
- multi_exp_vec_BC_model
- multi_exp_expE_vec_BC_model
- multi_exp_Asqr_vec_BC_model
- multi_exp_Asqr_expE_vec_BC_model
- multi_alt_exp_vec_BC_model
- multi_alt_exp_expE_vec_BC_model
- multi_alt_exp_Asqr_vec_BC_model
- multi_alt_exp_Asqr_expE_vec_BC_model

and require one further key in addition to the keys of the underlying models: the name of the constant corresponding to the temporal extent (of the lattice):

| Key | content | type |
|---|---|---|
| T_name | T | string |

The models with periodic boundary conditions have the same parameters as the underlying models. The only difference is the replacement

$$f_i(\mathtt{t}) \quad \rightarrow \quad f_i(\mathtt{t}) + f_i(\mathtt{T} - \mathtt{t})$$

for all functions $f_i$ of the model. The value of T has to be specified in the constant_values node; see section 9.

## 13.4 Two-point models with time-independent contributions

For all the “scalar” and “vector” two-point models listed in 13.1 and 13.2, as well as their versions with periodic boundary conditions (Sec. 13.3), an additional version exists, which adds time-independent pieces to the fit function:

$$f(\mathtt{t}) \quad \rightarrow \quad f(\mathtt{t}) + \mathtt{C}$$

for scalar models,

$$f(\mathtt{t}) \quad \rightarrow \quad f(\mathtt{t}) + \mathtt{C} + (-1)^{\mathtt{t}+1}\mathtt{Co}$$

for scalar models with oscillating contributions,

$$f_i(\mathtt{t}) \quad \rightarrow \quad f_i(\mathtt{t}) + \mathtt{C\_\_}i$$

for vector models, and

$$f_i(\mathtt{t}) \quad \rightarrow \quad f_i(\mathtt{t}) + \mathtt{C\_\_}i + (-1)^{\mathtt{t}+1}\mathtt{Co\_\_}i$$

for vector models with oscillating contributions. The quantities C, Co, $\{\mathtt{C\_\_}i\}$, $\{\mathtt{Co\_\_}i\}$ (as appropriate) are additional fit parameters. These models are called

- multi_exp_const_model
- multi_exp_expE_const_model

- multi_exp_Asqr_const_model
- multi_exp_Asqr_expE_const_model
- multi_alt_exp_const_model
- multi_alt_exp_expE_const_model
- multi_alt_exp_Asqr_const_model
- multi_alt_exp_Asqr_expE_const_model
- multi_exp_vec_const_model
- multi_exp_expE_vec_const_model
- multi_exp_Asqr_vec_const_model
- multi_exp_Asqr_expE_vec_const_model
- multi_alt_exp_vec_const_model
- multi_alt_exp_expE_vec_const_model
- multi_alt_exp_Asqr_vec_const_model
- multi_alt_exp_Asqr_expE_vec_const_model
- multi_exp_BC_const_model
- multi_exp_expE_BC_const_model
- multi_exp_Asqr_BC_const_model
- multi_exp_Asqr_expE_BC_const_model
- multi_alt_exp_BC_const_model
- multi_alt_exp_expE_BC_const_model
- multi_alt_exp_Asqr_BC_const_model
- multi_alt_exp_Asqr_expE_BC_const_model
- multi_exp_vec_BC_const_model
- multi_exp_expE_vec_BC_const_model
- multi_exp_Asqr_vec_BC_const_model
- multi_exp_Asqr_expE_vec_BC_const_model
- multi_alt_exp_vec_BC_const_model
- multi_alt_exp_expE_vec_BC_const_model
- multi_alt_exp_Asqr_vec_BC_const_model

- `multi_alt_exp_Asqr_expE_vec_BC_const_model`

and require one further key in addition to the keys of the underlying models: the name template for the new parameter(s):

| Key | content | type |
|---|---|---|
| `C_name` | `C` | string |

## 13.5 Matrix two-point models

Matrix models are very different from vector models. In matrix models, it is assumed that the amplitudes factor into an outer product of a vector with itself, like $\texttt{A\_\_}i\ \texttt{A\_\_}j$, where the $\texttt{A\_\_}i$ are used as fit parameters.

In the following, the functions are labelled by two indices $i$, $j$. The required storage order in the data files is such that the first index ($i$) runs slow and the second index ($j$) runs fast.

### 13.5.1 `multi_exp_mat_model`

- function(s): for $i = 1...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$f_{ij}(\texttt{t}) = \texttt{A\_\_}i\ \texttt{A\_\_}j\ \left[ e^{-\texttt{E t}} + \sum_{n=1}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(\texttt{E+dE\_1+}...\texttt{+dE\_}n)\texttt{t}} \right]$$

- variable(s): `t`
- parameter(s): $\{\texttt{A\_\_}i\}$, $\{\texttt{B\_}n\texttt{\_\_}i\}$ (for $i = 1...\max(\texttt{dim\_1}, \texttt{dim\_2})$), `E`, $\{\texttt{dE\_}n\}$
- properties:

| Key | content | type |
|---|---|---|
| `n_exp` | $N$ | integer $\geq 1$ |
| `A_name` | `A` | string |
| `B_name` | `B` | string |
| `E_name` | `E` | string |
| `dE_name` | `dE` | string |
| `t_name` | `t` | string |
| `dim_1` | $i = 1...\texttt{dim\_1}$ | integer $\geq 1$ |
| `dim_2` | $j = 1...\texttt{dim\_2}$ | integer $\geq 1$ |

### 13.5.2 `multi_exp_expE_mat_model`

- function(s): for $i = 1...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$f_{ij}(\texttt{t}) = \texttt{A\_\_}i\ \texttt{A\_\_}j\ \left[ e^{-e^{\texttt{E}}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+...+e^{\texttt{dE\_}n})\texttt{t}} \right]$$

- variable(s), parameter(s), properties: same as `multi_exp_mat_model`

### 13.5.3 `multi_alt_exp_mat_model`

- function(s): for $i = 1...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$
\begin{aligned}
f_{ij}(\texttt{t}) \;=\; & \texttt{A\_\_}i\ \texttt{A\_\_}j\ \left[ e^{-\texttt{E}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+\texttt{dE\_1}+...+\texttt{dE\_}n)\texttt{t}} \right] \\
& +(-1)^{\texttt{t}+1}\texttt{Ao\_\_}i\ \texttt{Ao\_\_}j\ \left[ e^{-\texttt{Eo}\ \texttt{t}} + \sum_{m=1}^{M-1} \texttt{Bo\_}m\texttt{\_\_}i\ \texttt{Bo\_}m\texttt{\_\_}j\ e^{-(\texttt{Eo}+\texttt{dEo\_1}+...+\texttt{dEo\_}m)\texttt{t}} \right]
\end{aligned}
$$

- variable(s): `t`
- parameter(s):
  $\{\texttt{A\_\_}i\}$, $\{\texttt{B\_}n\texttt{\_\_}i\}$ (for $i = 1...\max(\texttt{dim\_1}, \texttt{dim\_2})$), `E`, $\{\texttt{dE\_}n\}$,
  $\{\texttt{Ao\_\_}i\}$, $\{\texttt{Bo\_}m\texttt{\_\_}i\}$ (for $i = 1...\max(\texttt{dim\_1}, \texttt{dim\_2})$), `Eo`, $\{\texttt{dEo\_}m\}$,
- properties:

| Key | content | type |
|---|---|---|
| `n_exp` | $N$ | integer $\geq 1$ |
| `n_o_exp` | $M$ | integer $\geq 1$ |
| `A_name` | `A` | string |
| `B_name` | `B` | string |
| `E_name` | `E` | string |
| `dE_name` | `dE` | string |
| `t_name` | `t` | string |
| `dim_1` | $i = 1...\texttt{dim\_1}$ | integer $\geq 1$ |
| `dim_2` | $j = 1...\texttt{dim\_2}$ | integer $\geq 1$ |

### 13.5.4 `multi_alt_exp_expE_mat_model`

- function(s): for $i = 1...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$
\begin{aligned}
f_{ij}(\texttt{t}) \;=\; & \texttt{A\_\_}i\ \texttt{A\_\_}j\ \left[ e^{-e^{\texttt{E}}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+...+e^{\texttt{dE\_}n})\texttt{t}} \right] \\
& +(-1)^{\texttt{t}+1}\texttt{Ao\_\_}i\ \texttt{Ao\_\_}j\ \left[ e^{-e^{\texttt{Eo}}\texttt{t}} + \sum_{m=1}^{M-1} \texttt{Bo\_}m\texttt{\_\_}i\ \texttt{Bo\_}m\texttt{\_\_}j\ e^{-(e^{\texttt{Eo}}+e^{\texttt{dEo\_1}}+...+e^{\texttt{dEo\_}m})\texttt{t}} \right]
\end{aligned}
$$

- variable(s), parameter(s), properties: same as `multi_alt_exp_mat_model`

## 13.6 Matrix two-point models, type II

In type II matrix models, the ground state is not special. All amplitudes, including the ground-state amplitude, are written as a product $\texttt{A\_\_}i\ \texttt{B\_}n\texttt{\_\_}i$ (i.e., $n$ now starts from 0). This means that $\max(\texttt{dim\_1}, \texttt{dim\_2})$ of the parameters $\{\texttt{B\_}n\texttt{\_\_}i\}$ are redundant, and Bayesian constraints must be activated. The typical usage is to constrain the parameters $\texttt{B\_}(i-1)\texttt{\_\_}i$ to $1 \pm \epsilon$ with a very small prior width $\epsilon$, which effectively eliminates these parameters from the functions.

#### 13.6.1 `multi_exp_mat_II_model`

- function(s): for $i = 1...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$f_{ij}(\texttt{t}) = \texttt{A\_\_}i\ \texttt{A\_\_}j \sum_{n=0}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+...+\texttt{dE\_}n)\texttt{t}}$$

- variable(s): `t`
- parameter(s): $\{\texttt{A\_\_}i\}$, $\{\texttt{B\_}n\texttt{\_\_}i\}$ (for $i = 1...\max(\texttt{dim\_1}, \texttt{dim\_2})$), `E`, $\{\texttt{dE\_}n\}$
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| dim_1 | $i = 1...\texttt{dim\_1}$ | integer $\geq 1$ |
| dim_2 | $j = 1...\texttt{dim\_2}$ | integer $\geq 1$ |

#### 13.6.2 `multi_exp_expE_mat_II_model`

- function(s): for $i = 1...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$f_{ij}(\texttt{t}) = \texttt{A\_\_}i\ \texttt{A\_\_}j \sum_{n=0}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+...+e^{\texttt{dE\_}n})\texttt{t}}$$

- variable(s), parameter(s), properties: same as `multi_exp_mat_II_model`

### 13.7 Triangular matrix two-point models

These models are like matrix models with `dim_1=dim_2`, but the triangular models consist of only the functions with $j \geq i$. This is intended for matrix fits with exactly symmetric (i.e. symmetrized) data.

#### 13.7.1 `multi_exp_mat_upper_model`

- function(s): for $i = 1...\texttt{dim}$, $j = i...\texttt{dim}$ (total number of functions $= \texttt{dim}(\texttt{dim}+1)/2$):

$$f_{ij}(\texttt{t}) = \texttt{A\_\_}i\ \texttt{A\_\_}j \left[ e^{-\texttt{E t}} + \sum_{n=1}^{N-1} \texttt{B\_}n\texttt{\_\_}i\ \texttt{B\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+\texttt{dE\_1}+...+\texttt{dE\_}n)\texttt{t}} \right]$$

- variable(s): `t`
- parameter(s): $\{\texttt{A\_\_}i\}$, $\{\texttt{B\_}n\texttt{\_\_}i\}$ (for $i = 1...\texttt{dim}$), `E`, $\{\texttt{dE\_}n\}$
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| dim | $i = 1...\mathtt{dim}$, $j = i...\mathtt{dim}$ | integer $\geq 1$ |

### 13.7.2 multi_exp_expE_mat_upper_model

- function(s): for $i = 1...\mathtt{dim}$, $j = i...\mathtt{dim}$ (total number of functions = $\mathtt{dim}(\mathtt{dim}+1)/2$):

$$f_{ij}(\mathtt{t}) = \mathtt{A\_\_}i\ \mathtt{A\_\_}j\ \left[ e^{-e^{\mathtt{E}}\ \mathtt{t}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\mathtt{\_\_}i\ \mathtt{B\_}n\mathtt{\_\_}j\ e^{-(e^{\mathtt{E}}+e^{\mathtt{dE\_1}}+...+e^{\mathtt{dE\_}n})\mathtt{t}} \right]$$

- variable(s), parameter(s), properties: same as multi_exp_mat_upper_model

### 13.7.3 multi_exp_mat_II_upper_model

Note: some parameters are redundant (see Sec. 13.6).

- function(s): for $i = 1...\mathtt{dim}$, $j = i...\mathtt{dim}$ (total number of functions = $\mathtt{dim}(\mathtt{dim}+1)/2$):

$$f_{ij}(\mathtt{t}) = \mathtt{A\_\_}i\ \mathtt{A\_\_}j \sum_{n=0}^{N-1} \mathtt{B\_}n\mathtt{\_\_}i\ \mathtt{B\_}n\mathtt{\_\_}j\ e^{-(\mathtt{E}+...+\mathtt{dE\_}n)\mathtt{t}}$$

- variable(s): t
- parameter(s): $\{\mathtt{A\_\_}i\}$, $\{\mathtt{B\_}n\mathtt{\_\_}i\}$ (for $i = 1...\max(\mathtt{dim\_1}, \mathtt{dim\_2})$), E, $\{\mathtt{dE\_}n\}$
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| dim | $i = 1...\mathtt{dim}$, $j = i...\mathtt{dim}$ | integer $\geq 1$ |

### 13.7.4 multi_exp_expE_mat_II_upper_model

Note: some parameters are redundant (see Sec. 13.6).

- function(s): for $i = 1...\mathtt{dim}$, $j = i...\mathtt{dim}$ (total number of functions = $\mathtt{dim}(\mathtt{dim}+1)/2$):

$$f_{ij}(\mathtt{t}) = \mathtt{A\_\_}i\ \mathtt{A\_\_}j \sum_{n=0}^{N-1} \mathtt{B\_}n\mathtt{\_\_}i\ \mathtt{B\_}n\mathtt{\_\_}j\ e^{-(e^{\mathtt{E}}+...+e^{\mathtt{dE\_}n})\mathtt{t}}$$

- variable(s), parameter(s), properties: same as multi_exp_mat_II_upper_model

## 13.8 Non-symmetric matrix two-point models

Here the amplitudes are factorized into an outer product of *two different vectors*, rather than the outer product of a vector with itself, as in the models of Sec. 13.5. Because of a reparametrization invariance, some amplitude parameters need to be eliminated to get unique results. This has already been done in the following models, so that the fit functions are different for $i = 1$ vs $i > 1$ (see below).

As in Sec. 13.5, the required storage order in the data files is such that the first index ($i$) runs slow and the second index ($j$) runs fast.

### 13.8.1 multi_exp_nonsym_mat_model

- function(s):

  for $i = 2...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$f_{ij}(\texttt{t}) = \texttt{Ax\_\_}i\ \texttt{Ay\_\_}j\ \left[e^{-\texttt{E}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{Bx\_}n\texttt{\_\_}i\ \texttt{By\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+\texttt{dE\_1}+...+\texttt{dE\_}n)\texttt{t}}\right]$$

  for $i = 1$, $j = 1...\texttt{dim\_2}$:

$$f_{1j}(\texttt{t}) = \texttt{Ay\_\_}j\ \left[e^{-\texttt{E}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{By\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+\texttt{dE\_1}+...+\texttt{dE\_}n)\texttt{t}}\right]$$

- variable(s): t
- parameter(s): {$\texttt{Ax\_\_}i$, $\texttt{Bx\_}n\texttt{\_\_}i$} (for $i = 2...\texttt{dim\_1}$), {$\texttt{Ay\_\_}j$, $\texttt{By\_}n\texttt{\_\_}j$} (for $j = 1...\texttt{dim\_2}$), E, {$\texttt{dE\_}n$}
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| dim_1 | $i = 1...\texttt{dim\_1}$ | integer $\geq 2$ |
| dim_2 | $j = 1...\texttt{dim\_2}$ | integer $\geq 1$ |

### 13.8.2 multi_exp_expE_nonsym_mat_model

- function(s):

  for $i = 2...\texttt{dim\_1}$, $j = 1...\texttt{dim\_2}$:

$$f_{ij}(\texttt{t}) = \texttt{Ax\_\_}i\ \texttt{Ay\_\_}j\ \left[e^{-e^{\texttt{E}}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{Bx\_}n\texttt{\_\_}i\ \texttt{By\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+...+e^{\texttt{dE\_}n})\texttt{t}}\right]$$

for $i = 1$, $j = 1 ... \texttt{dim\_2}$:

$$f_{1j}(\texttt{t}) = \texttt{Ay\_\_}j \left[ e^{-e^{\texttt{E}}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{By\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+...+e^{\texttt{dE\_}n})\texttt{t}} \right]$$

- variable(s), parameter(s), properties: same as `multi_exp_nonsym_mat_model`

#### 13.8.3 `multi_alt_exp_nonsym_mat_model`

- function(s):

  for $i = 2 ... \texttt{dim\_1}$, $j = 1 ... \texttt{dim\_2}$:

$$\begin{aligned} f_{ij}(\texttt{t}) &= \texttt{Ax\_\_}i\ \texttt{Ay\_\_}j \left[ e^{-\texttt{E}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{Bx\_}n\texttt{\_\_}i\ \texttt{By\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+\texttt{dE\_1}+...+\texttt{dE\_}n)\texttt{t}} \right] \\ &\quad +(-1)^{\texttt{t}+1}\ \texttt{Aox\_\_}i\ \texttt{Aoy\_\_}j \left[ e^{-\texttt{Eo}\ \texttt{t}} + \sum_{m=1}^{M-1} \texttt{Box\_}m\texttt{\_\_}i\ \texttt{Boy\_}m\texttt{\_\_}j\ e^{-(\texttt{Eo}+\texttt{dEo\_1}+...+\texttt{dEo\_}m)\texttt{t}} \right] \end{aligned}$$

  for $i = 1$, $j = 1 ... \texttt{dim\_2}$:

$$\begin{aligned} f_{1j}(\texttt{t}) &= \texttt{Ay\_\_}j \left[ e^{-\texttt{E}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{By\_}n\texttt{\_\_}j\ e^{-(\texttt{E}+\texttt{dE\_1}+...+\texttt{dE\_}n)\texttt{t}} \right] \\ &\quad +(-1)^{\texttt{t}+1}\ \texttt{Aoy\_\_}j \left[ e^{-\texttt{Eo}\ \texttt{t}} + \sum_{m=1}^{M-1} \texttt{Boy\_}m\texttt{\_\_}j\ e^{-(\texttt{Eo}+\texttt{dEo\_1}+...+\texttt{dEo\_}m)\texttt{t}} \right] \end{aligned}$$

- variable(s): `t`

- parameter(s): {Ax__$i$, Bx_$n$__$i$} (for $i = 2 ... \texttt{dim\_1}$), {Ay__$j$, By_$n$__$j$} (for $j = 1 ... \texttt{dim\_2}$), E, {dE_$n$}, {Aox__$i$, Box_$m$__$i$} (for $i = 2 ... \texttt{dim\_1}$), {Aoy__$j$, Boy_$m$__$j$} (for $j = 1 ... \texttt{dim\_2}$), Eo, {dEo_$n$}

- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| n_o_exp | $M$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| dim_1 | $i = 1 ... \texttt{dim\_1}$ | integer $\geq 2$ |
| dim_2 | $j = 1 ... \texttt{dim\_2}$ | integer $\geq 1$ |

#### 13.8.4 `multi_alt_exp_expE_nonsym_mat_model`

- function(s):

  for $i = 2 ... \texttt{dim\_1}$, $j = 1 ... \texttt{dim\_2}$:

$$
\begin{aligned}
f_{ij}(\texttt{t}) \;=\;& \texttt{Ax\_\_}i\ \texttt{Ay\_\_}j\ \left[ e^{-e^{\texttt{E}}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{Bx\_}n\texttt{\_\_}i\ \texttt{By\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+...+e^{\texttt{dE\_}n})\texttt{t}} \right] \\
&+(-1)^{\texttt{t+1}}\, \texttt{Aox\_\_}i\ \texttt{Aoy\_\_}j\ \left[ e^{-e^{\texttt{Eo}}\ \texttt{t}} + \sum_{m=1}^{M-1} \texttt{Box\_}m\texttt{\_\_}i\ \texttt{Boy\_}m\texttt{\_\_}j\ e^{-(e^{\texttt{Eo}}+e^{\texttt{dEo\_1}}+...+e^{\texttt{dEo\_}m})\texttt{t}} \right]
\end{aligned}
$$

  for $i = 1$, $j = 1 ... \texttt{dim\_2}$:

$$
\begin{aligned}
f_{1j}(\texttt{t}) \;=\;& \texttt{Ay\_\_}j\ \left[ e^{-e^{\texttt{E}}\ \texttt{t}} + \sum_{n=1}^{N-1} \texttt{By\_}n\texttt{\_\_}j\ e^{-(e^{\texttt{E}}+e^{\texttt{dE\_1}}+...+e^{\texttt{dE\_}n})\texttt{t}} \right] \\
&+(-1)^{\texttt{t+1}}\, \texttt{Aoy\_\_}j\ \left[ e^{-e^{\texttt{Eo}}\ \texttt{t}} + \sum_{m=1}^{M-1} \texttt{Boy\_}m\texttt{\_\_}j\ e^{-(e^{\texttt{Eo}}+e^{\texttt{dEo\_1}}+...+e^{\texttt{dEo\_}m})\texttt{t}} \right]
\end{aligned}
$$

- variable(s), parameter(s), properties: same as `multi_alt_exp_nonsym_mat_model`

### 13.9 Scalar three-point models

#### 13.9.1 `threept_multi_exp_model`

- function(s):

$$
f(\texttt{t},\,\texttt{T}) = \texttt{A}\left[ e^{-\texttt{F}\ \texttt{t}} e^{-\texttt{E(T-t)}} + \sum_{\substack{n\ = 0\,...\,N\ -1,\\ n' = 0\,...\,N'-1,\\ (n,n')\neq(0,0)}} \texttt{B\_}n'\texttt{\_}n\ e^{-(\texttt{F+dF\_1+...+dF\_}n')\texttt{t}}\ e^{-(\texttt{E+dE\_1+...+dE\_}n)(\texttt{T-t})} \right]
$$

- variable(s): `t`, `T`
- parameter(s): `A`, $\{\texttt{B\_}n'\texttt{\_}n\}$, `E`, $\{\texttt{dE\_}n\}$, `F`, $\{\texttt{dF\_}n'\}$
- properties:

| Key | content | type |
|---|---|---|
| n_exp_initial | $N$ | integer $\geq 1$ |
| n_exp_final | $N'$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_initial_name | E | string |
| dE_initial_name | dE | string |
| E_final_name | F | string |
| dE_final_name | dF | string |
| t_name | t | string |
| T_name | T | string |

### 13.9.2 threept_multi_exp_expE_model

- function(s):

$$f(\mathtt{t},\ \mathtt{T}) = \mathtt{A}\left[e^{-e^{\mathtt{F}}\mathtt{t}}e^{-e^{\mathtt{E}}(\mathtt{T}-\mathtt{t})} + \sum_{\substack{n\ = 0\,...\,N\ -1,\\ n' = 0\,...\,N'-1,\\ (n,n')\neq(0,0)}} \mathtt{B}\_n'\_n\ e^{-(e^{\mathtt{F}}+e^{\mathtt{dF\_1}}+...+e^{\mathtt{dF\_}n'})\mathtt{t}}\ e^{-(e^{\mathtt{E}}+e^{\mathtt{dE\_1}}+...+e^{\mathtt{dE\_}n})(\mathtt{T}-\mathtt{t})}\right]$$

- variable(s), parameter(s), properties: same as threept_multi_exp_model

### 13.9.3 threept_multi_alt_exp_model

- function(s):

For $M > 0$ and $M' > 0$:

$$
\begin{aligned}
f(\mathtt{t},\,\mathtt{T}) \quad = \quad & \mathtt{Aee}\left[e^{-\mathtt{F}\ \mathtt{t}}e^{-\mathtt{E}(\mathtt{T}-\mathtt{t})} + \sum_{\substack{n\ = 0\,...\,N\ -1,\\ n' = 0\,...\,N'-1,\\ (n,n')\neq(0,0)}} \text{Bee\_}n'\text{\_}n\ e^{-(\mathtt{F}+\mathtt{dF\_1}+...+\mathtt{dF\_}n')\mathtt{t}}\ e^{-(\mathtt{E}+\mathtt{dE\_1}+...+\mathtt{dE\_}n)(\mathtt{T}-\mathtt{t})}\right] \\
+ \quad (-1)^t \quad & \mathtt{Aoe}\left[e^{-\mathtt{Fo}\ \mathtt{t}}e^{-\mathtt{E}(\mathtt{T}-\mathtt{t})} + \sum_{\substack{n\ = 0\,...\,N\ -1,\\ m' = 0\,...\,M'-1,\\ (n,m')\neq(0,0)}} \text{Boe\_}m'\text{\_}n\ e^{-(\mathtt{Fo}+\mathtt{dFo\_1}+...+\mathtt{dFo\_}m')\mathtt{t}}\ e^{-(\mathtt{E}+\mathtt{dE\_1}+...+\mathtt{dE\_}n)(\mathtt{T}-\mathtt{t})}\right] \\
+\ (-1)^{(T-t)}\ & \mathtt{Aeo}\left[e^{-\mathtt{F}\ \mathtt{t}}e^{-\mathtt{Eo}(\mathtt{T}-\mathtt{t})} + \sum_{\substack{m\ = 0\,...\,M\ -1,\\ n' = 0\,...\,N'-1,\\ (m,n')\neq(0,0)}} \text{Beo\_}n'\text{\_}m\ e^{-(\mathtt{F}+\mathtt{dF\_1}+...+\mathtt{dF\_}n')\mathtt{t}}\ e^{-(\mathtt{Eo}+\mathtt{dEo\_1}+...+\mathtt{dEo\_}m)(\mathtt{T}-\mathtt{t})}\right] \\
+ \quad (-1)^T \quad & \mathtt{Aoo}\left[e^{-\mathtt{Fo}\ \mathtt{t}}e^{-\mathtt{Eo}(\mathtt{T}-\mathtt{t})} + \sum_{\substack{m\ = 0\,...\,M\ -1,\\ m' = 0\,...\,M'-1,\\ (m,m')\neq(0,0)}} \text{Boo\_}m'\text{\_}m\ e^{-(\mathtt{Fo}+\mathtt{dFo\_1}+...+\mathtt{dFo\_}m')\mathtt{t}}\ e^{-(\mathtt{Eo}+\mathtt{dEo\_1}+...+\mathtt{dEo\_}m)(\mathtt{T}-\mathtt{t})}\right]
\end{aligned}
$$

**Note:** for $M' = 0$, the second and fourth row disappear.
for $M = 0$, the third and fourth row disappear.

- variable(s): `t`, `T`
- parameter(s):

  `Aee`, {Bee_$n'$_$n$}, `E`, {`dE`_$n$}, `F`, {`dF`_$n'$}

  For $M > 0$ also `Aeo`, {Beo_$n'$_$m$}, `Eo`, {`dEo`_$m$}

  For $M' > 0$ also `Aoe`, {Boe_$m'$_$n$}, `Fo`, {`dFo`_$m'$}

  For ($M > 0$ and $M' > 0$) also `Aoo`, {Boo_$m'$_$m$}

- properties:

| Key | content | type |
|---|---|---|
| n_exp_initial | $N$ | integer $\geq 1$ |
| n_o_exp_initial | $M$ | integer $\geq 0$ |
| n_exp_final | $N'$ | integer $\geq 1$ |
| n_o_exp_final | $M'$ | integer $\geq 0$ |
| A_name | A | string |
| B_name | B | string |
| E_initial_name | E | string |
| dE_initial_name | dE | string |
| E_final_name | F | string |
| dE_final_name | dF | string |
| t_name | t | string |
| T_name | T | string |

#### 13.9.4 threept_multi_alt_exp_expE_model

- function: same as threept_multi_alt_exp_model, but with the following replacements:

$$\begin{aligned}
\texttt{E} &\rightarrow e^{\texttt{E}} \\
\texttt{dE_}n &\rightarrow e^{\texttt{dE_}n} \\
\texttt{F} &\rightarrow e^{\texttt{F}} \\
\texttt{dF_}n' &\rightarrow e^{\texttt{dF_}n'} \\
\texttt{Eo} &\rightarrow e^{\texttt{Eo}} \\
\texttt{dEo_}m &\rightarrow e^{\texttt{dEo_}m} \\
\texttt{Fo} &\rightarrow e^{\texttt{Fo}} \\
\texttt{dFo_}m' &\rightarrow e^{\texttt{dFo_}m'}
\end{aligned}$$

- variable(s), parameter(s), properties: same as threept_multi_alt_exp_model

### 13.10 Vector three-point models

The "scalar" three-point models listed in section 13.9 are also available as "vector" three-point models, called

- threept_multi_exp_vec_model
- threept_multi_exp_expE_vec_model
- threept_multi_alt_exp_vec_model
- threept_multi_alt_exp_expE_vec_model

These models require one further key in addition to the keys of the corresponding scalar models: the dimension of the vector:

| Key | content | type |
|---|---|---|
| dim | $i = 1...\texttt{dim}$ | integer $\geq 1$ |

A vector model has then `dim` functions $f_i(\mathtt{t})$ ($i = 1...\mathtt{dim}$) of the same form as the underlying scalar model. These functions have individual amplitude parameters, but share all the energy parameters. For example, the functions for `threept_multi_exp_vec_model` are

$$f_i(\mathtt{t},\ \mathtt{T}) = \mathtt{A\_\_}i \left[ e^{-\mathtt{F}\ \mathtt{t}} e^{-\mathtt{E}(\mathtt{T}-\mathtt{t})} + \sum_{\substack{n\ = 0\,...\,N\ - 1,\\ n' = 0\,...\,N' - 1,\\ (n,n') \neq (0,0)}} \mathtt{B\_}n'\mathtt{\_}n\mathtt{\_\_}i\ e^{-(\mathtt{F}+\mathtt{dF\_1}+...+\mathtt{dF\_}n')\mathtt{t}}\ e^{-(\mathtt{E}+\mathtt{dE\_1}+...+\mathtt{dE\_}n)(\mathtt{T}-\mathtt{t})} \right]$$

for $i = 1...\mathtt{dim}$.

## 13.11 "Degenerate" three-point models

### 13.11.1 `threept_constant_model`

- function(s):

$$f(\mathtt{t},\ \mathtt{T}) = \mathtt{C}$$

- variable(s): `t`, `T` (note: both `t` and `T` are a dummy variables.)
- parameter(s): `C`
- properties:

| Key | content | type |
|---|---|---|
| `C_name` | `C` | string |
| `t_name` | `t` | string |
| `T_name` | `T` | string |

### 13.11.2 `threept_constant_sqr_model`

- function(s):

$$f(\mathtt{t},\ \mathtt{T}) = \mathtt{C}^2$$

- variable(s), parameter(s), properties: same as `threept_constant_model`

### 13.11.3 `multi_exp_2var_model`

- function(s):

$$f(\mathtt{t},\mathtt{T}) = \mathtt{A} \left[ e^{-\mathtt{E}\ \mathtt{T}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\ e^{-(\mathtt{E}+\mathtt{dE\_1}+...+\mathtt{dE\_}n)\mathtt{T}} \right]$$

- variable(s): `t`, `T` (note: `t` is a dummy variable. This model is intended for fitting of three-point functions in which the initial and the final state have identical energy levels, and the off-diagonal transition matrix elements vanish. In this case, the `t`-dependence disappears.)

- parameter(s): A, {B_$n$}, E, {dE_$n$}
- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| T_name | T | string |

#### 13.11.4 multi_exp_expE_2var_model

- function(s):

$$f(\mathtt{t},\mathtt{T}) = \mathtt{A}\left[e^{-e^{\mathtt{E}}\ \mathtt{T}} + \sum_{n=1}^{N-1} \mathtt{B\_}n\ e^{-(e^{\mathtt{E}}+e^{\mathtt{dE\_1}}+\ldots+e^{\mathtt{dE\_}n})\mathtt{T}}\right]$$

- variable(s), parameter(s), properties: same as multi_exp_2var_model

### 13.12 Multi-particle two-point models

#### 13.12.1 multi_part_exp_expE_model

This model was implemented by W. Detmold. Currently, it has no implementation of the symbolic derivatives, and therefore this model must be used in combination with

```
<num_diff_first_order> true </num_diff_first_order>
```

in the fit_settings node (see Sec. 12) to enable numerical differentiation.

- function(s):

$$\begin{aligned} f(\mathtt{t},\mathtt{T}) &= \mathtt{Z\_}P\ e^{-e^{\mathtt{E\_}P}(\mathtt{T}/2)}\ \cosh(e^{\mathtt{E\_}P}(\mathtt{t}-\mathtt{T}/2)) \\ &+ \sum_{p=1}^{\lfloor P/2 \rfloor} 2\binom{P}{p}\ \mathtt{Z\_}P\ \mathtt{A\_}P\mathtt{\_}p\ e^{-(e^{\mathtt{E\_}(P-p)}+e^{\mathtt{E\_}p})(\mathtt{T}/2)}\ \cosh\left[(e^{\mathtt{E\_}(P-p)}+e^{\mathtt{E\_}p})(\mathrm{t}-\mathtt{T}/2)\right] \\ &+ \sum_{n=1}^{N-1} \mathtt{Z\_}P\ \mathtt{B\_}P\mathtt{\_}n\ e^{-(e^{\mathtt{E\_}P}+e^{\mathtt{dE\_}P\mathtt{\_1}}+\ldots+e^{\mathtt{dE\_}P\mathtt{\_}n})(\mathtt{T}/2)}\ \cosh\left[(e^{\mathtt{E\_}P}+e^{\mathtt{dE\_}P\mathtt{\_1}}+\ldots+e^{\mathtt{dE\_}P\mathtt{\_}n})(\mathrm{t}-\mathtt{T}/2)\right] \end{aligned}$$

- variable(s): t
- constant(s): T
- parameter(s): Z_$P$, {A_$P$_$p$}, {B_$P$_$n$}, {E_$p$}, {dE_$n$}

- properties:

| Key | content | type |
|---|---|---|
| n_exp | $N$ | integer $\geq 1$ |
| n_part | $P$ | integer $\geq 1$ |
| A_name | A | string |
| B_name | B | string |
| Z_name | Z | string |
| E_name | E | string |
| dE_name | dE | string |
| t_name | t | string |
| T_name | T | string |

## 14 User-defined model

In addition to the built-in models listed in Sec. 13, there is a model called

parse_model,

which allows the user to completely define new models using only the XML input file. When using parse_model, the functions (and, if necessary their first-order derivatives) are entered as strings and parsed by XMBF. A parse_model of XMBF offers the same functionality as a user-defined model of QMBF, allowing the definition of models with an arbitrary number of functions, variables, parameters, and constants. The functions (and derivatives) can be constructed using the same elementary operations as in QMBF (the reader is referred to the QMBF manual for the details):

```
+, -, *, /,
(, ),
exp(...), log(...),
sin(...), cos(...), tan(...),
sinh(...), cosh(...), tanh(...),
arcsin(...), arccos(...), arctan(...),
sqr(...), sqrt(...),
alt(...).
```

An example usage of `parse_model` is shown below:

```
<parse_model>

  <n_variables> 1 </n_variables>
  <n_functions> 2 </n_functions>

  <variables>
    <variable>
      <number> 1 </number>
      <name> t </name>
    </variable>
  </variables>

  <functions>
    <function>
      <number> 1 </number>
      <definition> A_1*(exp(-E*t)+exp(-E*(T-t))) </definition>
    </function>
    <function>
      <number> 2 </number>
      <definition> A_2*(exp(-E*t)+exp(-E*(T-t))) </definition>
    </function>
  </functions>

  <constants>
    <name> T </name>
  </constants>

  <parameters>
    <name> A_2 </name>
    <name> E </name>
    <name> A_1 </name>
  </parameters>

  <derivatives>
    <derivative>
      <function_number> 1  </function_number>
      <parameter_name> A_1  </parameter_name>
      <definition> exp(-E*t)+exp(-E*(T-t)) </definition>
    </derivative>
    <derivative>
      <function_number> 1  </function_number>
      <parameter_name> A_2  </parameter_name>
      <definition> 0 </definition>
    </derivative>
    <derivative>
      <function_number> 1 </function_number>
      <parameter_name> E </parameter_name>
      <definition> A_1*((-t)*exp(-E*t)-(T-t)*exp(-E*(T-t))) </definition>
    </derivative>
    <derivative>
      <function_number> 2 </function_number>
      <parameter_name> A_1 </parameter_name>
      <definition> 0 </definition>
    </derivative>
    <derivative>
      <function_number> 2 </function_number>
      <parameter_name> A_2 </parameter_name>
```

```
          <definition> exp(-E*t)+exp(-E*(T-t)) </definition>
        </derivative>
        <derivative>
          <function_number> 2 </function_number>
          <parameter_name> E </parameter_name>
          <definition> A_2*((-t)*exp(-E*t)-(T-t)*exp(-E*(T-t))) </definition>
        </derivative>
      </derivatives>

      <fit_domain>
        <variable_name> t </variable_name>
        <range>
          <min> 10 </min>
          <max> 54 </max>
        </range>
      </fit_domain>

      <data_file>
        <file_type> ASCII </file_type>
        <file_name> correlator.dat </file_name>
      </data_file>

    </parse_model>
```

(this particular example actually reimplements the case of `multi_exp_vec_BC_model` with `n_exp`=1 and `dim`=2).

Note that the `derivatives` node in `parse_model` is only required if the option `num_diff_first_order` in the `fit_settings` node (see Sec. 12) is set to `false`.