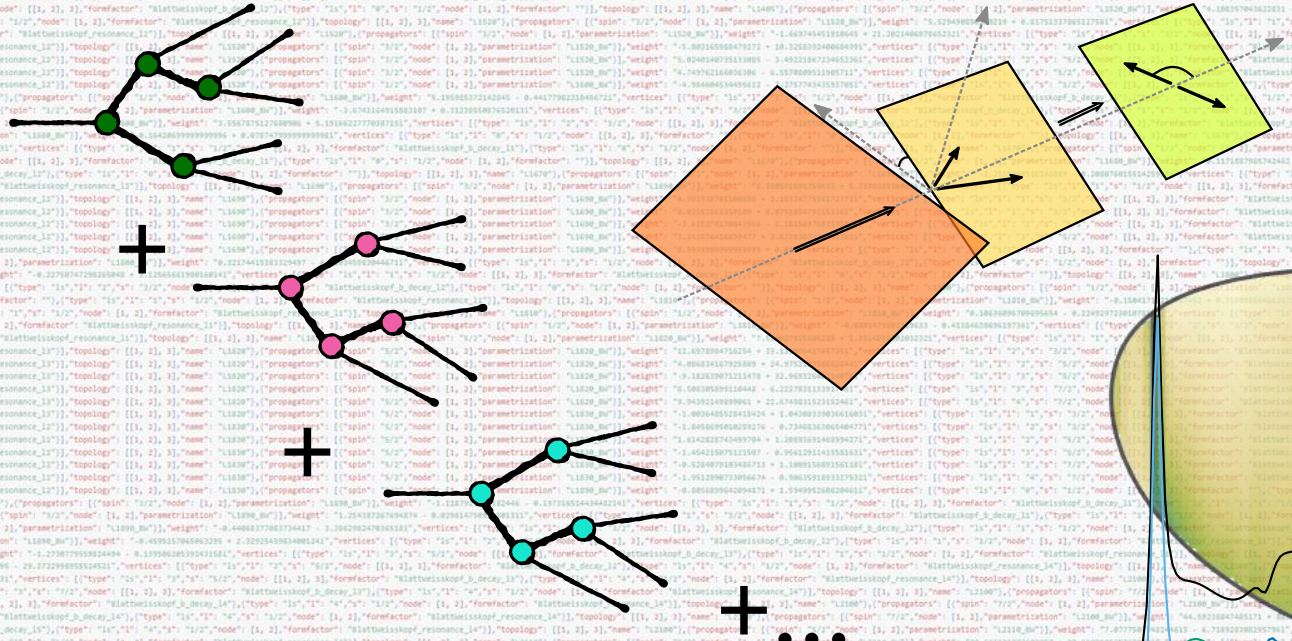


RUHR-UNIVERSITÄT BOCHUM

Model format for Hadronic Cascade Decays

"Democratizing Models" Workshop

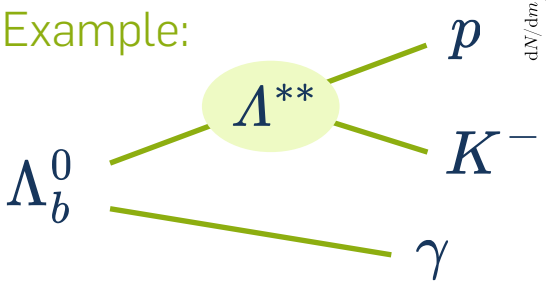
November 25th, 2024

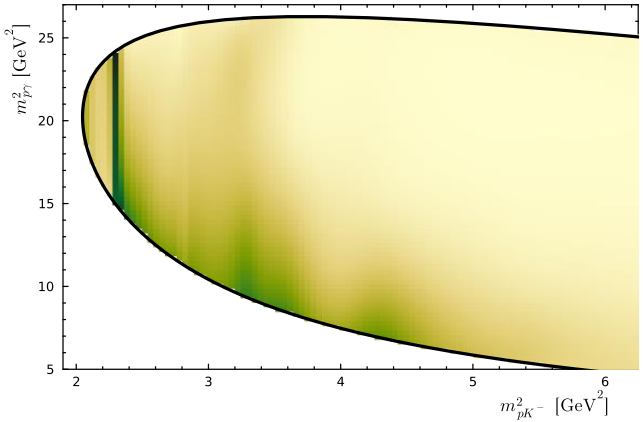
Remco de Boer

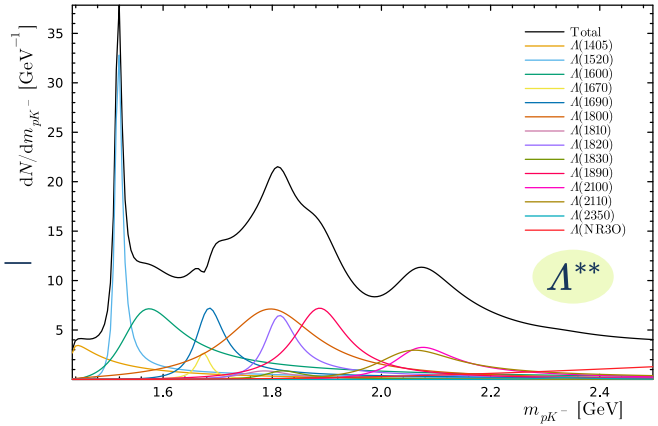
Hadronic cascade decays

- Interest: **intermediate states** in particle reaction
- Data input: four-momenta
  ⇒ **multi-dimensional data input**
- Strategy: construct models to describe measured **intensity distributions**
- Cascade decay model is most common (helicity formalism)

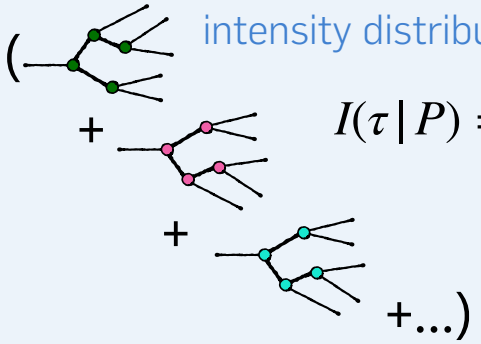
Example:







Hadronic cascade decays

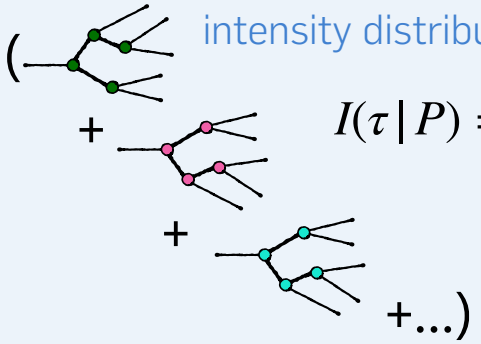


intensity distribution (PDF) $\mathbb{R}$

$$I(\tau \mid P) = \sum_{\lambda_0,\lambda_0'} \rho_{\lambda_0,\lambda_0'}(P) \sum_{\{\lambda_i\}}^{i!=0} A^*_{\lambda_0';\{\lambda\}} A_{\lambda_0;\{\lambda\}}$$

$$A_{\{\lambda\}}(\tau) = \sum_i A^i_{\{\lambda\}}(\tau)$$

Hadronic cascade decays

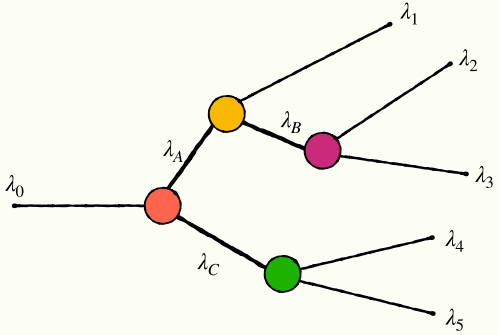


intensity distribution (PDF) $\mathbb{R}$

$$I(\tau \mid P) = \sum_{\lambda_0,\lambda_0'} \rho_{\lambda_0,\lambda_0'}(P) \sum_{\{\lambda_i\}}^{i!=0} A^*_{\lambda_0';\{\lambda\}} A_{\lambda_0;\{\lambda\}}$$

$$A_{\{\lambda\}}(\tau) = \sum_i A^i_{\{\lambda\}}(\tau)$$

complex-valued amplitude functions $\mathbb{C}$



$$A^i_{\lambda_0;\lambda_1,\lambda_2,\lambda_3,\lambda_4,\lambda_5} =$$

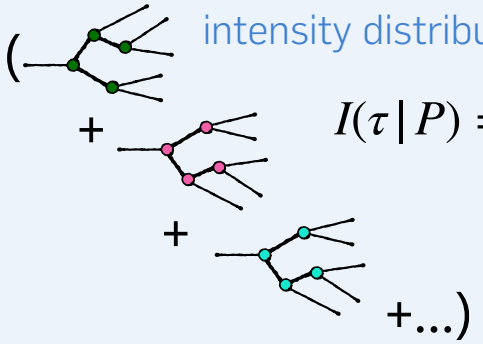
$$D^{j_0*}_{\lambda_0,\lambda_A-\lambda_C} H_{\lambda_A,\lambda_C} \; D^{j_A*}_{\lambda_A,\lambda_1-\lambda_B} H_{\lambda_1,\lambda_B} \; D^{j_B*}_{\lambda_B,\lambda_2-\lambda_3} H_{\lambda_2,\lambda_3} \; D^{j_C*}_{\lambda_C,\lambda_4-\lambda_5} H_{\lambda_4,\lambda_5}$$

$$\times P_A(s_A)\, P_B(s_B)\, P_C(s_C)$$

$$\times D^{j_1}_{\lambda_1',\lambda_1}(R_1^w)\, D^{j_2}_{\lambda_2',\lambda_2}(R_2^w)\, D^{j_3}_{\lambda_3',\lambda_3}(R_3^w)\, D^{j_4}_{\lambda_4',\lambda_4}(R_4^w)\, D^{j_5}_{\lambda_5',\lambda_5}(R_5^w)$$

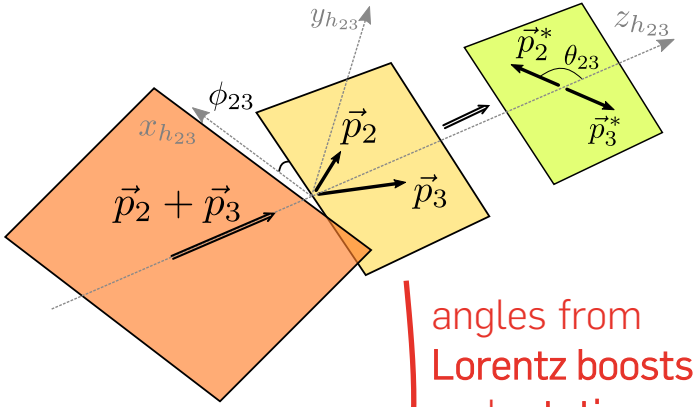
Hadronic cascade decays

intensity distribution (PDF) $\mathbb{R}$



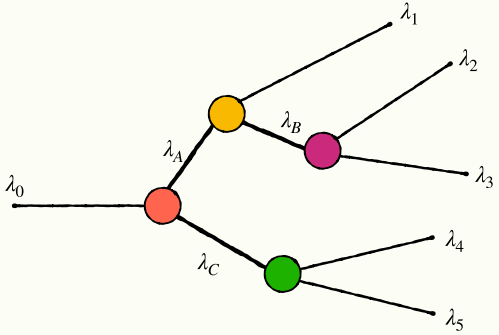
$$I(\tau \mid P) = \sum_{\lambda_0,\lambda_0'} \rho_{\lambda_0,\lambda_0'}(P) \sum_{\{\lambda_i\}}^{i!=0} A^*_{\lambda_0';\{\lambda\}} A_{\lambda_0;\{\lambda\}}$$

$$A_{\{\lambda\}}(\tau) = \sum_i A^i_{\{\lambda\}}(\tau)$$



angles from **Lorentz boosts** and **rotations**

complex-valued amplitude functions



$$A^i_{\lambda_0;\lambda_1,\lambda_2,\lambda_3,\lambda_4,\lambda_5} = \quad \mathbb{C}$$

$$D^{j_0*}_{\lambda_0,\lambda_A-\lambda_C} H_{\lambda_A,\lambda_C} \; D^{j_A*}_{\lambda_A,\lambda_1-\lambda_B} H_{\lambda_1,\lambda_B} \; D^{j_B*}_{\lambda_B,\lambda_2-\lambda_3} H_{\lambda_2,\lambda_3} \; D^{j_C*}_{\lambda_C,\lambda_4-\lambda_5} H_{\lambda_4,\lambda_5}$$

$$\times P_A(s_A)\, P_B(s_B)\, P_C(s_C)$$

$$\times D^{j_1}_{\lambda_1',\lambda_1}(R_1^w)\, D^{j_2}_{\lambda_2',\lambda_2}(R_2^w)\, D^{j_3}_{\lambda_3',\lambda_3}(R_3^w)\, D^{j_4}_{\lambda_4',\lambda_4}(R_4^w)\, D^{j_5}_{\lambda_5',\lambda_5}(R_5^w)$$

How to serialize?

- Requires extension of HS3 format
  - Complex-valued functions
  - PDFs composed of large, multidimensional functions
  - Dozens or hundreds of input parameters
  - Normalisation only numerically
- Encode information about decay chain
- Section for model validation

Current format specification

```
{
  "distributions": [ ···
  ],
  "functions": [ ···
  ],
  "domains": [ ···
  ],
  "misc": { ···
  },
  "parameter_points": [ ···
  ]
}
```

github.com/RUB-EP1/amplitude-serialization

Current format specification

- **`"distributions"`**
  - Recipe for constructing PDF sum
  - Decay description (kinematics)
  - Chain definitions with selected parametrization functions

github.com/RUB-EP1/amplitude-serialization

Current format specification

- **`"functions"`**
  - Dynamics parametrizations
    - **`"type"`** assumes the reader knows how to implement the function
    - Custom functions allowed (à la **`TFormula`** string)
  - Function input (data columns)
  - Parameter values

```
"functions": [
  {
    "name": "L1405_Flatte",
    "type": "MultichannelBreitWigner",
    "x": "m_12_sq",
    "mass": 1.405,
    "channels": [
      {
        "gsq": 0.24941478752959237,
        "ma": 0.938,
        "mb": 0.493,
        "l": 0,
        "d": 0
      },
      {…
      }
    ]
  },
  {…
  },
  {
    "type": "BlattWeisskopf",
    "l": 6,
    "radius": 5.0,
    "name": "BlattWeisskopf_b_decay_l6"
  }
],
```

github.com/RUB-EP1/amplitude-serialization

Current format specification

- **`"domains"`**
  - Variable definitions (names for the data input)
  - Expected value ranges
  - Used as arguments in **`"functions"`**

```
"domains": [
  {
    "name": "default",
    "type": "product_domain",
    "axes": [
      {
        "name": "cos_theta_12",
        "min": -1.0,
        "max": 1.0
      },
      {…
      },
      {…
      },
      {…
      },
      {…
      },
      {…
      },
      {…
      },
      {…
      }
    ]
  }
],
```

github.com/RUB-EP1/amplitude-serialization

Current format specification

Model validation:

- **`"misc"`**
  - “Checksums”: expected return values of functions
  - Any other metadata
- **`"parameter_points"`**
  Definition of validation points

```
"misc": {
  "amplitude_model_checksums": [
    {
      "point": "validation_point1",
      "distribution": "default_model",
      "value": 645.6877782510138
    },
    {…
    },
    {…
    }
  ]
},
"parameter_points": [
  {
    "name": "validation_point1",
    "parameters": [
      {
        "name": "cos_theta_12",
        "value": -0.2260000000000005
      },
      {…
      }
    ]
  },
```

github.com/RUB-EP1/amplitude-serialization

How to generalize?

- **Problems:**
  - Distribution description assumes sum recipe is known
  - Function type expected to be implemented by user
  - What about likelihood serialization?
- **Ideas:**
  - Full description of the function tree
  - Composable function definitions
  - Look for existing standards!
    e.g. neural network serialization