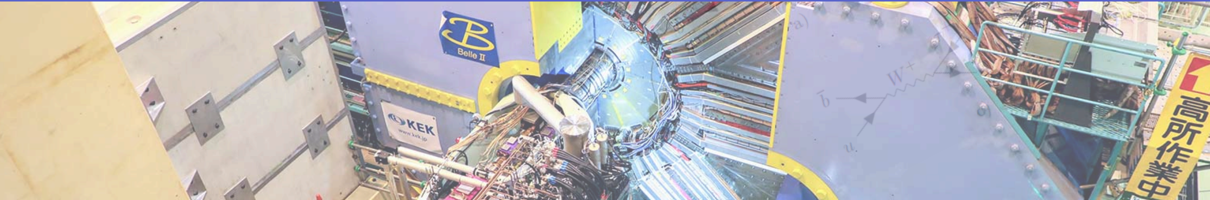
Search for the $B \to K\nu\bar{\nu}$ decay at Belle II using semileptonic tagging

Merna Abumusabh

Supervisors: Jerome Baudot, Giulio Dujany

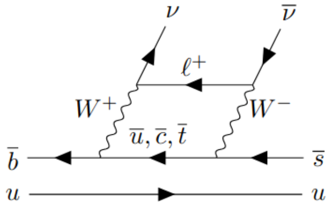
École doctorale Physique et chimie-physique

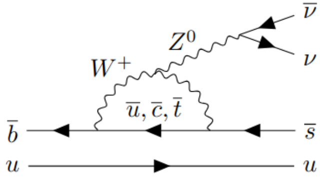
CSI – 2024-2025



Overview: Rare B-mesons Decays

Flavour-changing neutral currents (FCNCs) are rare processes in the Standard Model (SM), occurring only via loop-level diagrams like electroweak penguins or box diagrams. Their rarity makes them powerful tools to probe the SM's flavour structure and search for new physics. Potential contributions from heavy, beyond-SM particles in these loops could alter decay rates or affect final-state properties, offering indirect evidence for new phenomena.





Why Belle II?

- Clear environment with good particle identification.
- High statistics: Current integrated luminosity $\mathcal{L} = 365fb^{-1}$.

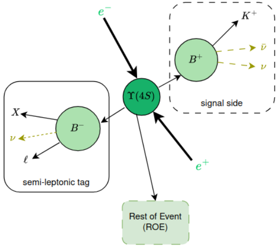
The $B \to K\nu\bar{\nu}$ Decay

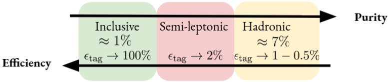
- The weak transition $b \to s\nu\bar{\nu}$ is characterized by significant missing energy due to neutrinos in the final state. It underlies the rare decay $B^+ \to K^+\nu\bar{\nu}$, which is highly suppressed in the Standard Model (SM), with a predicted branching ratio:

$$\mathcal{B}_{\mathrm{SM}}(B^+ \to K^+\nu\bar{\nu}) = (4.97 \pm 0.37) \times 10^{-6}.$$

- **Why $B \to K\nu\bar{\nu}$**:
  This channel is well described in the SM, which offers a clear path to investigate the BSM effects or validate the SM expected BF by calculating the signal strength:

$$\mu = \frac{\mathcal{B}(B \to K\nu\bar{\nu})_{\mathrm{measured}}}{\mathcal{B}(B \to K\nu\bar{\nu})_{\mathrm{SM}}}$$





Motivation

The $B \to K \nu \bar{\nu}$ decay has previously been studied using hadronic and inclusive tagging methods. This thesis focuses on exploring the decay using semi-leptonic tagging, providing a complementary approach to other Belle II analyses targeting this channel.

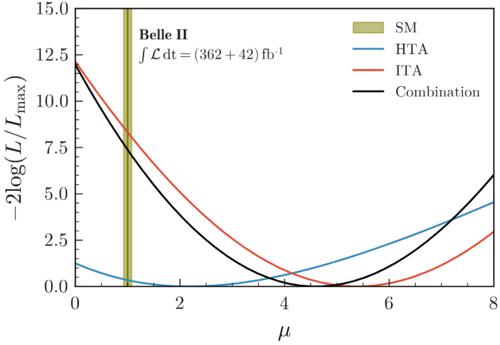


Figure 1: Evidence for $B^+ \to K^+ \nu \bar{\nu}$ decays[arXiv:2311.14647]

Training

A total of 86h ($\sim 80\%$) hours of training are done.

Disciplinary - 30h

- The Vlasov-Boltzmann Equation in Astrophysics : The dynamics of galaxies in equilibrium
- Un peu de géométrie et de topologie pour la physique et les physiciens Ens. et diff. de la culture scientifique

Transversal - 56h (Fully completed)

- Innovation course
- Congrès des doctorants ED 182 : participation AVEC présentation orale ou poster
- Identifying your skills: An introduction to the ”portfolio approach”
- ...

Upcoming summer school: The 2025 European School of High-Energy Physics 15-28 Oct, which will be added to the disciplinary hours.

The prompt Calibration at Belle II

- Due to the presence of many sub-detectors at Belle II, and the variation of their conditions over time, calibrations must be updated frequently to ensure accurate data reconstruction for physics analysis.
- The calibration at Belle II consists of two major steps: the prompt and the reprocessing. The prompt calibration provides the first set of calibrations for data, using short run ranges called ”buckets” that balance detector stability and sufficient statistics.

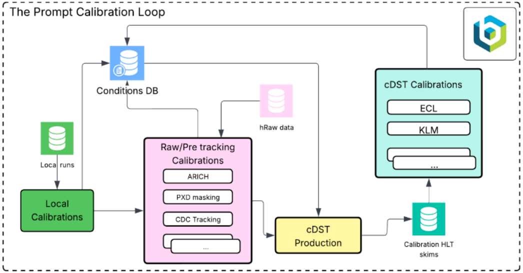


Figure 2: The main steps in the prompt calibration loop

The Prompt Calibration at Belle II

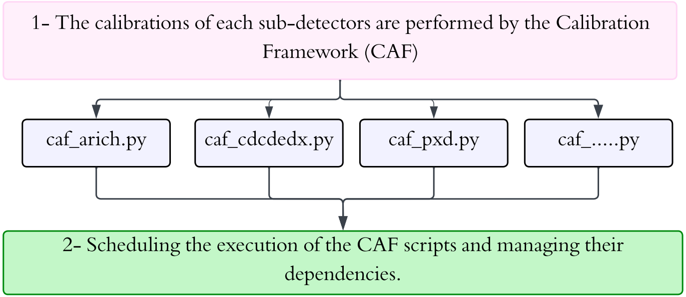


Figure 3: The oversight of the calibration

The Prompt Calibration at Belle II

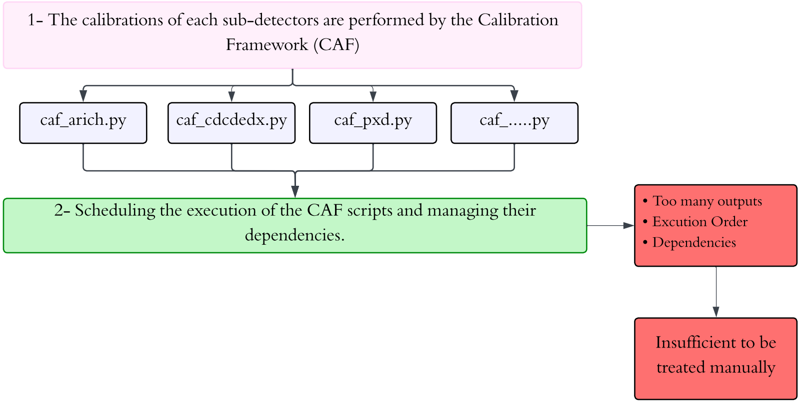


Figure 4: The oversight of the calibration

The Prompt Calibration at Belle II

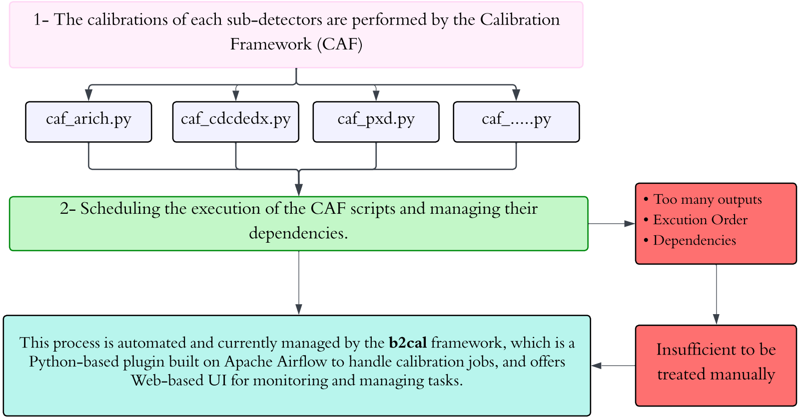


Figure 5: The oversight of the calibration

Service Task: Improving the automated prompt Calibration at Belle II

- **b2cal**, however, is a complex system with many interconnected components which introduced usability and maintenance challenges.
- **The service task was:**

Contributing to a prototype that will replace the current system and simplify prompt calibration implementation.

The (proposed) new prompt calibration framework: **b2luca** (**B2LU**igi* **CA**libration) .
The highlights of the b2luca design:

- Prioterizing simplicity: Similar functionalities to the current system, but in a much easier manner.
- No need for a web interface, Gitlab is used for the interaction.

Service Task: Improving the automated prompt Calibration at Belle II

The vast majority of the time during this period—up until last month—was dedicated to the service task.

- The new prototype is designed to address the complexity issues present in the current system.
- Most of the prototype has been developed and is undergoing continuous testing.
- Although the service task has been officially **validated**, active contributions to the prototype are still ongoing and will continue.
- A Belle II technical note is in preparation.

GraFEI Contribution

The Graph-based Full Event Interpretation (GraFEI) was presented at CHEP 2024 — a GNN-based model designed to inclusively reconstruct Belle II events. GraFEI predicts decay chains without prior assumptions, improving background rejection and signal efficiency. Its performance was evaluated in the context of the $B^+ \to K^+\nu\bar{\nu}$ search.

1. arXiv:2503.09401 [pdf, other] hep-ex physics.data-an

Graph-based Full Event Interpretation: a graph neural network for event reconstruction in Belle II

Authors: Merna Abumusabh, Jacopo Cerasoli, Giulio Dujany, Corentin Santos

Abstract: In this work we present the Graph-based Full Event Interpretation (GraFEI), a machine learning model based on graph neural networks to inclusively reconstruct events in the Belle~II experiment. Belle~II is well suited to perform measurements of B meson decays involving invisible particles (e.g. neutrinos) in the final state. The kinematical properties of such particles can be deduced from the en... ▽ More

Submitted 14 March, 2025; **v1** submitted 12 March, 2025; **originally announced** March 2025.

Comments: Proceedings for the 2024 Conference on Computing in High Energy and Nuclear Physics

"GraFEI: Graph-based Full Event Interpretation at Belle II," Proc. CHEP 2024, arXiv:2503.09401 [hep-ex].

Next Steps

- Contentious testing and improving b2luca.
- Going on with the analysis:
  - Finish generating the MC samples (in progress).
  - Event preselection criteria.
  - Train and deploy Boosted Decision Tree (BDT) for signal selection
  - Evaluate systematic uncertainties
  - Perform signal extraction fit to determine the signal strength μ

Thanks!

Back-ups


?
?
CALIBRATION
JOBS
JOBS
CALIBRATION
JOBS
JOBS
CALIBRATION
JOBS
JOBS
JOBS
DEPENTENCY

Airflow vs Luigi

Feature	Luigi	Airflow
Primary Use	ETL workflows	General orchestration, scheduling
Philosophy	Class-based dependencies	DAG-based scheduling
Dependencies	`requires()` method	Explicit chaining
I/O Tracking	Built-in with `Target`	XComs or external storage
Scheduler	`luigid` daemon	Advanced schedulers (e.g., Celery)
Parallelism	Limited	Scalable (Celery, Kubernetes)
Web UI	Basic task status view	Rich DAG, logs, Gantt charts
Monitoring	Minimal	Advanced SLAs, retries
Extensibility	Custom Python classes	Plugins, hooks, operators
Cloud Support	Limited (HDFS, Spark)	Broad (AWS, GCP, Docker)
Use Case	File-driven pipelines	Time/event-driven workflows

b2luigi: batch for Luigi

b2luigi is a workflow management framework built on top of Luigi, a Python-based system for handling complex workflows. It enables users to define and automate tasks by breaking them into separate, interdependent tasks.

The two main classes used in b2luigi are Task and WrapperTask.

- `Task`: Represents an individual unit of work with defined dependencies, outputs, and execution logic (run).
- `WrapperTask`: Groups multiple tasks together without executing any logic itself, ensuring all required tasks run before marking completion.

The main things to define in a Luigi Task are:

- Parameter: Equivalent to the python "constructors", and should be declared at the beginning of the task.
- output: Define the path and name of the output expected.
- run: The actual code that is being executed.

```python
import b2luigi
from termcolor import colored

class TaskA(b2luigi.Task):
    input_data = b2luigi.Parameter()

    def output(self):
        return b2luigi.LocalTarget("first_output.txt")

    def run(self):
        with open(self.input_data, "r") as f:
            inputs = f.read()

        with self.output().open("w") as ff:
            ff.write(inputs)
        print(colored(f"inputs are written to {self.output}"), "green")

if __name__ == "__main__":
    b2luigi.process([TaskA(input_data="input_data.txt")])
```

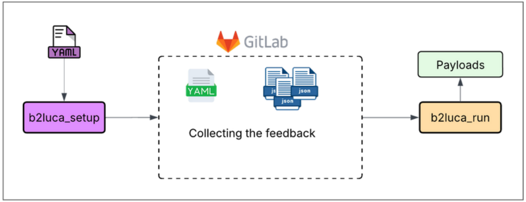
```
===== Luigi Execution Summary =====

Scheduled 1 tasks of which:
* 1 ran successfully:
    - 1 TaskA(input_data=input_data.txt)

This progress looks :) because there were no failed tasks or missing dependencies

===== Luigi Execution Summary =====
```

b2luca structure

b2luca is a command-line interface (CLI) framework developed to fully automate prompt calibration. It adopts a modular, configuration-driven architecture that enables rapid adaptation to evolving parameters .
The workflow in b2luca is divided into two main phases:

- A configuration phase focused on setting parameters.
- The execution of the calibration loop itself.

These phases are separated by a validation period during which calibration responsibles review and approve the settings.



The primary configuration file

- The yaml configuration file serves as the foundational input for b2luca’s execution, this file defines the critical paths for input and output, as well as the core parameters that govern the behavior of subsequent calibration tasks, such as the IoV, GTs, .. etc.
- Serves also as a “map” of b2luca, where you can locate where were the files created and saved.

```yaml
name: bucketXX

iov: [31, 1, 31, 22]

release: release-09-00-00
calibration_root_directory: "./calibration_test/cal_root"
run_mode: "Local" # "LSF", "PBS", "HTCondor"

staging_gt: &staging_gt data_prompt_rel08_staging
mcdr_gt: mcrd_prompt_rel08_staging

globaltags:
  - *staging_gt
  - data_prompt_rel08
  - online
  - patch_main_release-09

calibrations:

local_calibrations:
gitlab_repository:
  link: https://gitlab.desy.de/~/test-automated-calibration-tasks
  milestone: prompt_bucketXX
  local_path: "./calibration_test/git_repo"

hRaw_path: "/pnfs/sdcc.bnl.gov/data/bellediskdata/TMP/belle/hRaw/"
path_logs: "/pnfs/sdcc.bnl.gov/belleiicalib/test/"
BNL_path: "/pnfs/sdcc.bnl.gov/data/bellediskdata/TMP/belle/hRaw/"

cli_args: "--dry-run"
```

First step: `b2luca_setup`

`b2luca_setup` is the first step in prompt calibration, initializing a campaign by preparing configurations, settings, and GitLab issues for tracking the progress.
The setup step performs the following key operations:

- Fetches all calibration configs (local, raw, cDST) based on the specified basf2 version.
- Auto-generates GitLab issues for each task, including a summary tracker and task-to-issue ID map.
- Create a JSON dependency file for parallel task execution.

All those components, along with a copy of the configuration file, are committed to the GitLab repository specified within the configuration. These files are organized within a dedicated directory named after the calibration bucket.

Collecting the feedback

Each calibration responsible must review their assigned calibrations, ensure correct settings and payloads, make necessary adjustments, and resolve any issues before the running step.

Second Step: `b2luca_run`

1. Getting the Updated Settings: Retrieve latest config from the calibration repository, including verified settings and YAML updates from the calibration team.
2. 2- Generate Local Run Info CSV: Queries the online run DB using start/end experiment and run numbers from config.
3. Starts the calibration loop.

The Calibration Task

1. Creating a config file:The content of this file is parsed from the settings json files created in b2luca setup.
2. Creating an input data file: JSON file containing the path to the input data files with the assigned run and experiment.
3. Producing the calibration constants.
4. Creating validation scripts.
5. Running the Validation.
6. Logging the status on the assigned GitLab issue

Figure 7: The execution flow of the calibration task.