GLoBES

Patrick Huber

University of Wisconsin – Madison

1st GLoBES workshop 2007

Heidelberg, Germany, January 24, 2007

Outline

- Motivation
- Software requirements
- GLoBES history
- GLoBES milestones
- Design considerations
- New features
- Summary
- Outlook

Status Quo

Neutrino physics has been exceedingly interesting and vibrant the last 10 years. This very active phase was completely data driven. Flavor transitions were discovered in

- atmospheric neutrinos
- solar neutrinos
- reactor neutrinos
- accelerator neutrinos

This makes neutrino physics one of the most successful branches of high energy physics in this decades, so far.

The Future

In order to fully exploit and understand these exciting discoveries new experiments are needed.

- within a given technology – many open optimization issues
- between different technologies – evaluation and comparison
- global picture – robustness & synergies

The actual decisions will depend on a number of factors, many of them are of political or financial nature. However, all scientific arguments, ultimately, are based on a reliable and reproducible calculation of the physics sensitivity.

Requirements

- Reliability
- Reproducibility
- Flexibility
- Efficiency
- Documentation

Reliability

- Re-use of code, the more a code has been used in real world applications the less likely are severe bugs.
- Extensive testing
- Good documentation
- Intuitive API with error checking

Reproducibility

The information given a publication or proposal is not sufficient to reproduce the sensitivity estimates.

- General data storage and exchange format for the inputs $\Leftrightarrow$ flexibility?
- All implicit assumptions and approximations have to be documented, that includes the actual algorithms $\Leftrightarrow$ accuracy of documentation?
- Version control and archiving

Flexibility

Any system to promote reliability and reproducibility which is too rigid in daily use will be not used at all or by-passed.

- Developer user dialog to identify useful (and feasible) features
- API design

Flexibility quite often is difficult to reconcile with the other requirements.

Efficiency

The faster the code, the more thorough the analysis will be because more parameter studies can be performed

- physics parameters
- systematics parameters
- L-E
- . . .

Efficient code is the easier to write, the more specific the task is.

Documentation

Without good documentation, the best software is useless or will be after very short time (=memory decay constant of typical physicist). This is a general problem with legacy code!

Document what you do – do what you document

and make sure that the average user understands what is going on. Also documentation needs testing and debugging.

Open Source

An open source approach can fulfill many of these requirements quite naturally.

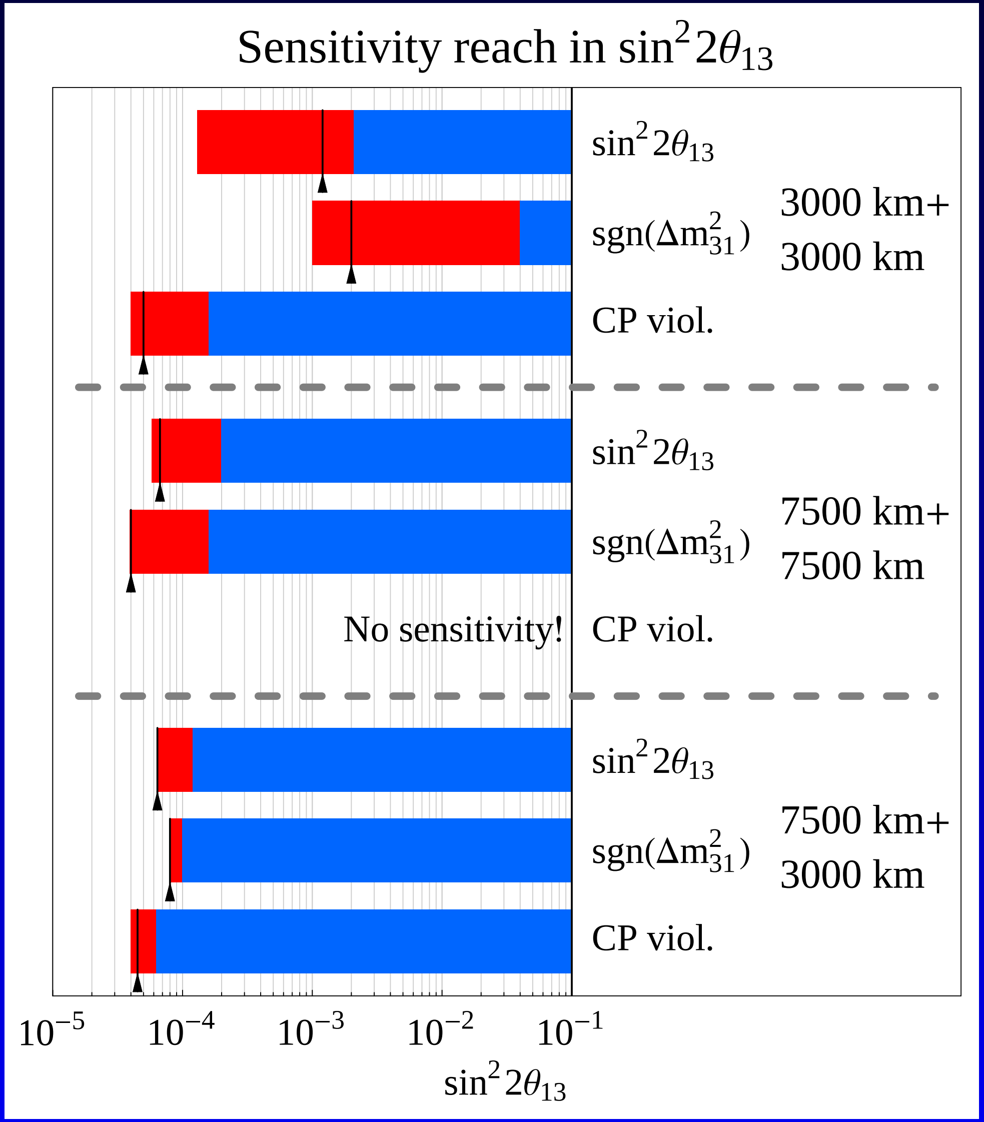
- Reliability – anyone can look into the source, anyone can use it
- Reproducibility – if you have the source and the inputs to a program, what more can you ask for?
- Flexibility – open source projects usually have more dialog between developers and users
- Efficiency – with the source anyone can optimize the code to his/her own needs.

GLoBES history

- development started 2004 – PH, M. Lindner, W. Winter
- major effort went into documentation
- first release August 2004 – version 2.0.0
- major bug fix release March 2005 version 2.0.11
- J. Kopp and M. Rolinec joined in July 2005
- 44 publications citing the GLoBES paper, creating a total of 377 citations

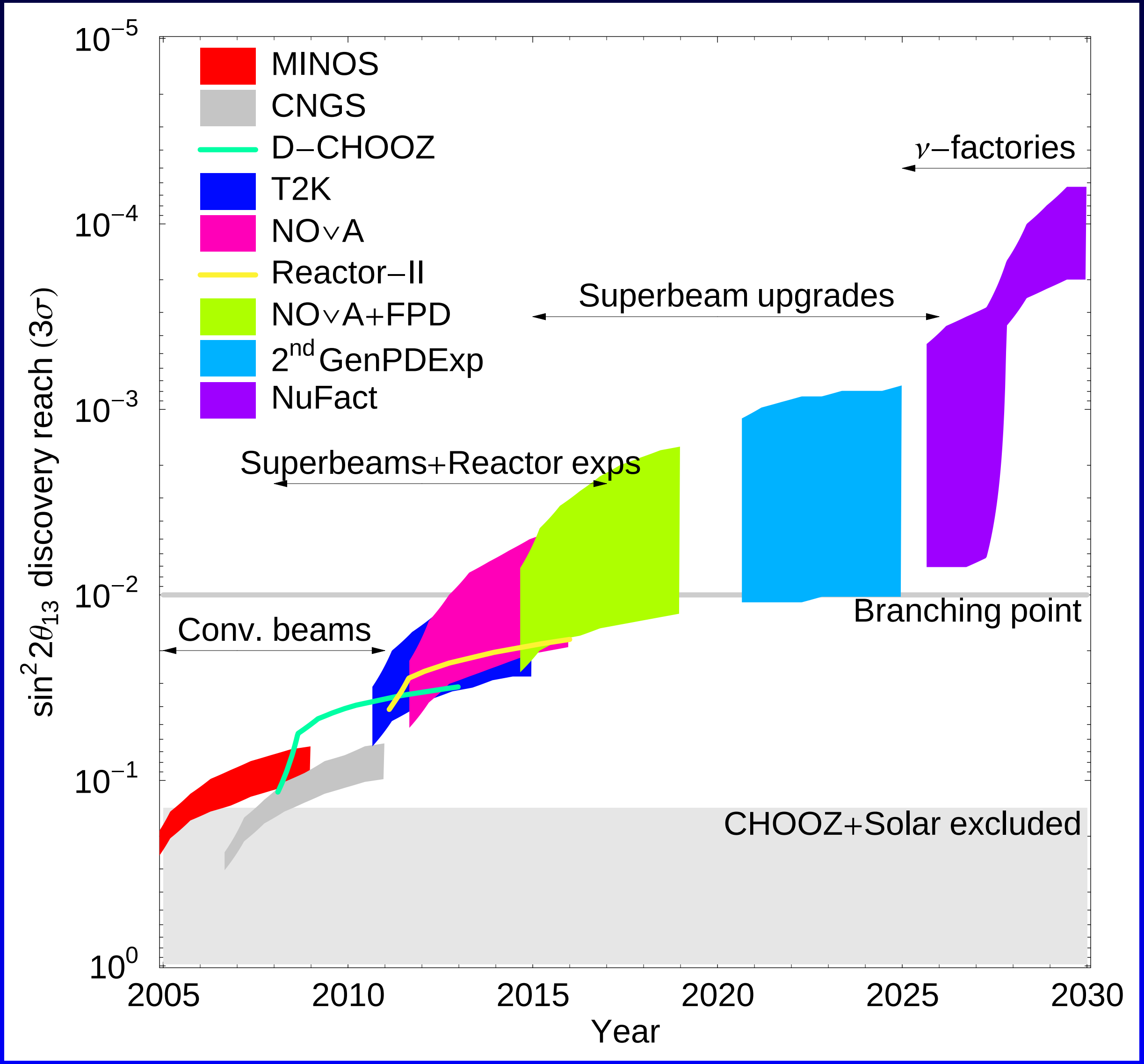
Milestones

APS study



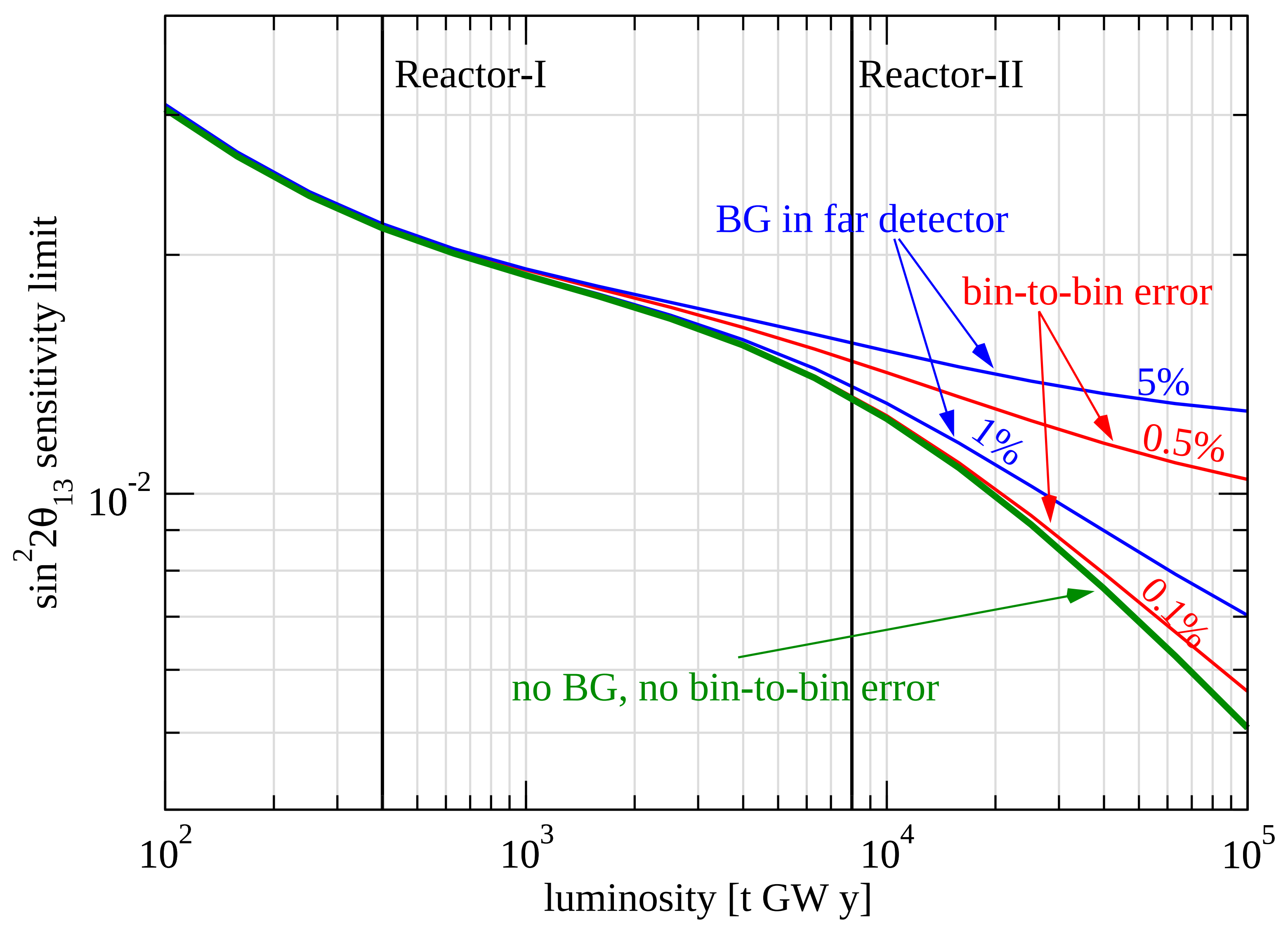
Milestones

Fermilab's Proton driver report



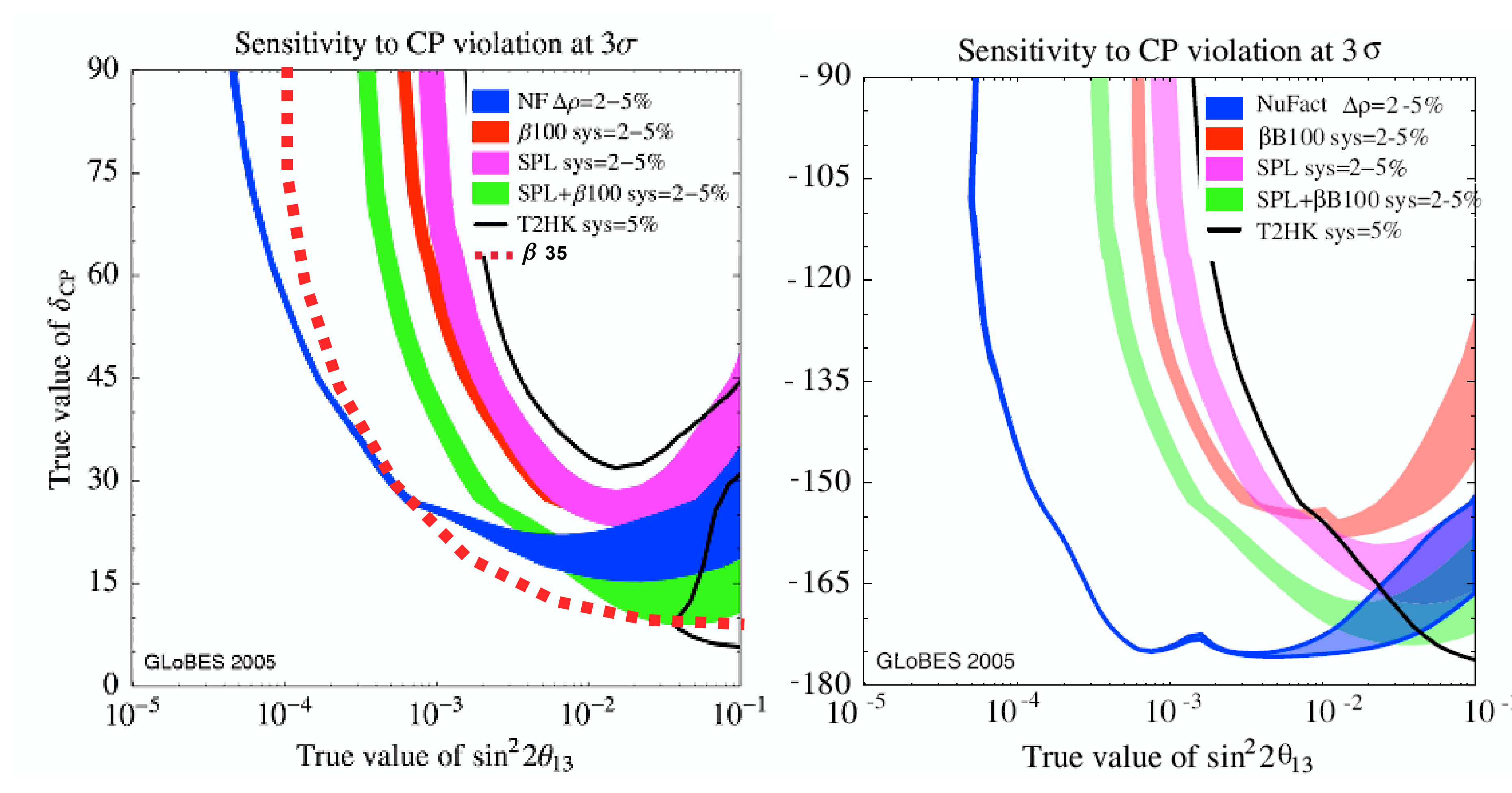
Milestones

White paper on reactor neutrinos



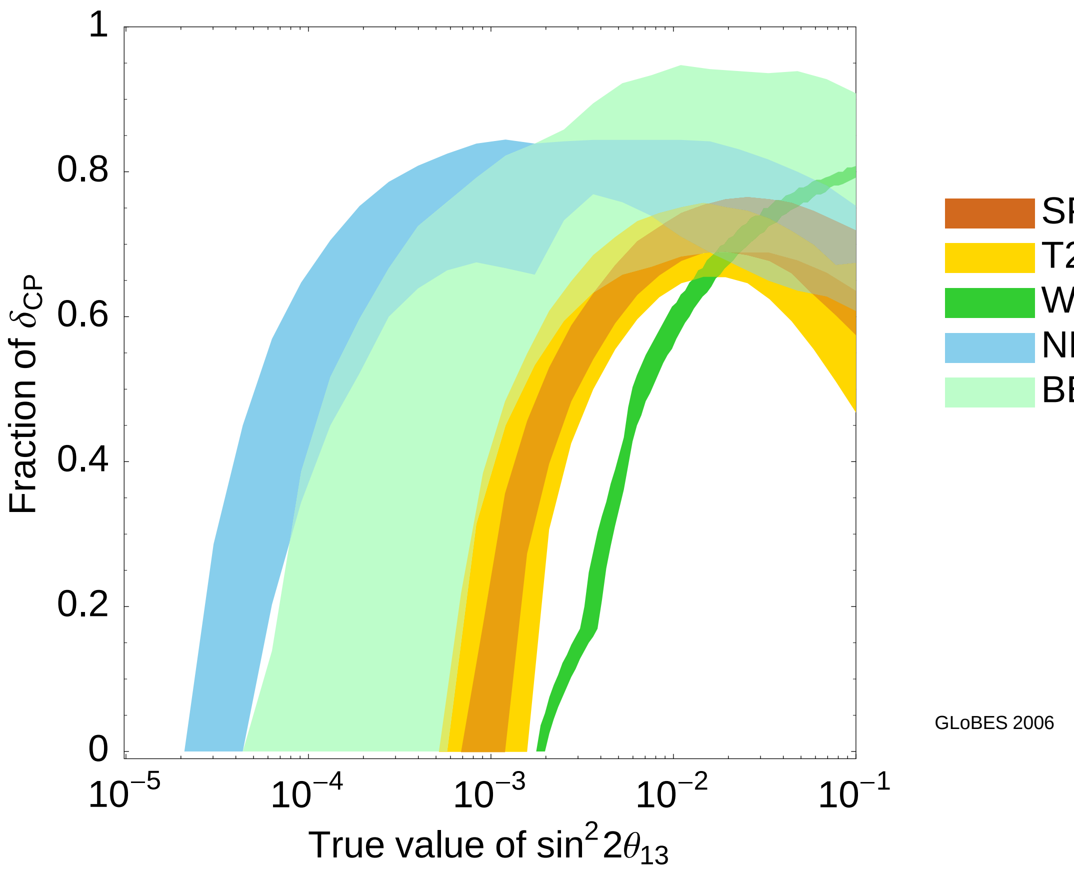
Milestones

CERN strategy group



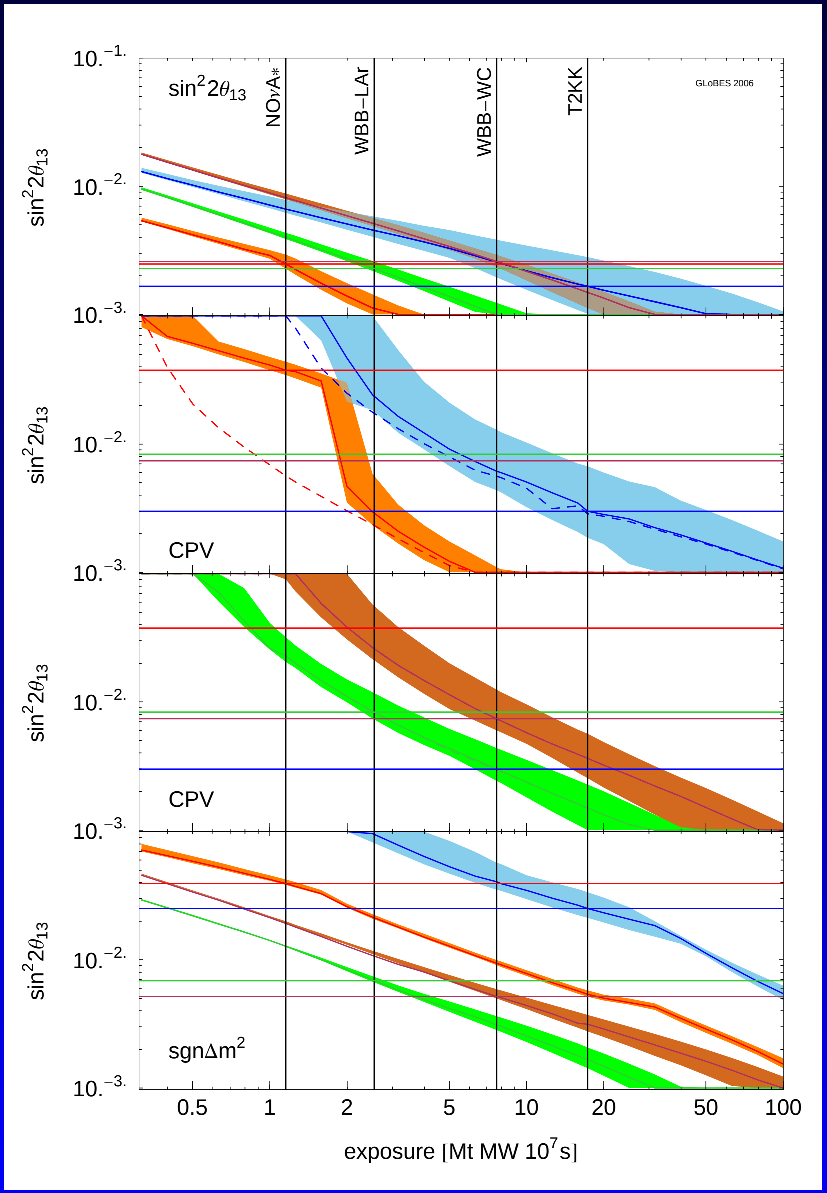
Milestones

ISS



Milestones

Joint BNL-FNAL study group (ongoing)



Milestones

- Double Chooz proposal
- BNL proposal
- . . .

Design considerations

- GPL
- C-library – very portable, easy to interface, numerically efficient
- Unix style separation of functionality – freedom to design analysis and to use any graphics tools
- Experiments are defined using AEDL – relatively complicated parser, transparent experiment definition
- Pull approach for systematics – flexible and intuitive
- Local minimization instead of grids – much faster

Features

- Accurate treatment of systematical errors
- Arbitrary matter profile & uncertainties
- Arbitrary energy resolution function
- Single and multiple experiment simulation
- Simple χ^2 calculation
- Inclusion of external input
- Projection of χ^2 (minimization)
- $\ldots$

Limitations

- systematics only rule by rule, e.g. no common normalization
- only one source and one detector setups
- only standard three flavor oscillation
- only parabolic priors
- AEDL does not support lists as variables
- relies on LAPACK

GLoBES has moved

It is developed and maintained by

- PH
- Joachim Kopp
- Manfred Lindner
- Mark Rolinec
- Walter Winter

and has moved to Heidelberg!
new URL – http://www.mpi-hd.mpg.de/lin/globes/

new email – globes@mpi-hd.mpg.de

What’s new in 3.0

New features

- user-defined systematics
- user-defined oscillation probability engine
- user-defined priors
- full support for lists in AEDL
- interpolating functions in AEDL
- fully updated glb-files

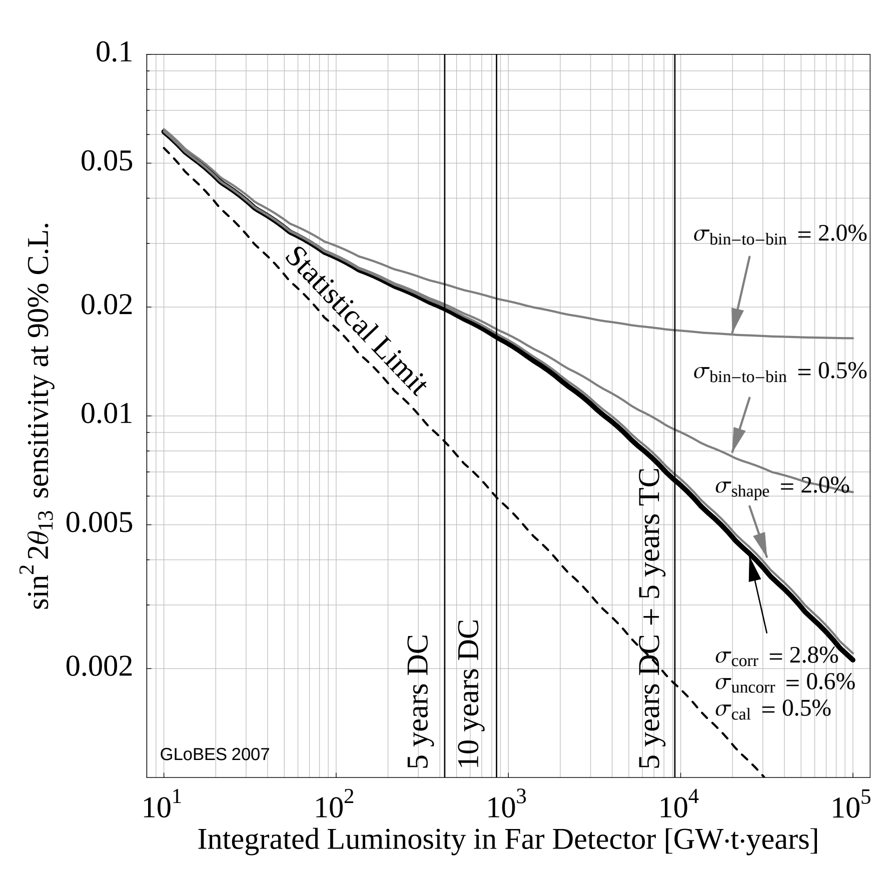
Internal changes

- custom built matrix diagonalization
- LAPACK no longer needed & C++ code removed
- new minimization scheme

User-defined systematics

This feature allows to simulate two detector setups like Double Chooz.

- define χ^2-function
- register it at run-time
- refer to it in AEDL by name

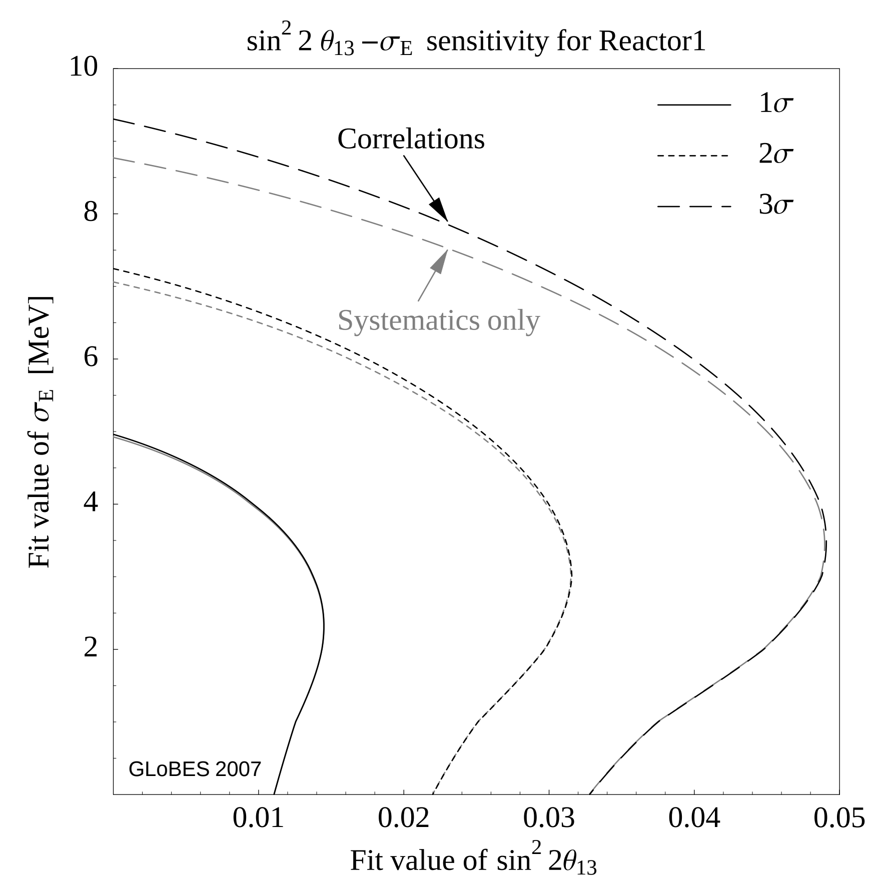


output of `example5`

User-defined oscillation engine

This feature allows to analyses non-standard physics scenarios like decoherence

- define oscillation engine
- register it at run-time
- use the new parameters
- can also be used to improve speed



output of `example6`

Improved AEDL

Interpolation allows easy, bin-independent definition of efficiencies, backgrounds etc.

```
/* ####### Energy dependent efficiencies ###### */

%posteffs={0.,1.,1.}

%energ={4.,20.,50.}

%bc=bincenter()

%inter=interpolation(%energ,%posteffs,1,%bc)
```

from `NFstandard.glb`

Additional: strict version control, `@norm` clarified

Summary

GLoBES

- is the only open source software of its kind
- has withstood the test of time
- is at the core of most strategy documents

GLoBES 3.0

- is now completely in C
- has added flexibility to deal with complex two detector setups and non-standard physics
- AEDL can now handle lists
- Speed has been greatly improved, due to a custom built matrix diagonalization

Future directions

Near term

- more different minimization schemes
- Python binding
- public mailing list
- anonymous CVS access

Long term

- Degeneracy finding
- AEDL migration to Perl
- Migration to C++