*P*roper

*A*nalysis of

*C*oherent

*E*xcitations

# An Overview of the PACE Project and its Software Product

## Outline of slides:

- PACE goals and motivation
- PACE outline of features
- PACE Project timeline and current status
    - Core data analysis framework
    - Model scattering function computation
    - Resolution convolution

UKRI Science and Technology Facilities Council ISIS

# PACE Goals

PACE is a major software project to provide:

- a single visualisation, simulation and fitting environment for the Excitations Group spectrometers
- for parallel and distributed computing.

Goals of PACE:

Expand and improve the existing Horace software suite for analysis of neutron scattering data, and integrate it more effectively with modelling codes to:

- Allow realistic simulations to be performed before and during the beam time, to make the most effective use of neutron beam
- Enable quantitative analysis of the full set of data collected during the experiment, with proper account of instrument resolution.
- Lower the barrier for users to analyse their data and to reduce the time between experiment and publication.

# PACE Motivation

LET and MERLIN create 4D maps of S(**Q**,ω) at the rate of 1 – 2 TB per day.

- Experiments can take years to analyse
- Many users are overwhelmed by the volume of data: major barrier for entry for new users
- Users must be expert and/or in close collaboration with expert instrument scientists

**Technical bottlenecks:**

- Most functions of HORACE are not parallelised and cannot operate on the latest data volumes.
- Modelling codes are not implemented for high performance computing, or their output is not integrated into the analysis framework.
  - Often need to write ad hoc 'home brew' code to interface with modelling code
  - Users want models on tap, or to interface simply to their own models
- Resolution of (single crystal or powder) data – is mostly not performed
  - Usually treated in ad hoc, simple-minded fashion
  - Proper implementation requires access to distributed computing or high performance computing to be feasible

# PACE Features (1/2)

PACE will build on the capabilities of the current Horace software for single crystal data analysis, and include and extend the features of powder analysis familiar to users of Mslice in Mantid or the earlier Matlab version of Mslice.

New PACE Features compared with Horace:

- Horace functionality for manipulating multidimensional datasets extended to parallel and distributed computing
  - includes unary (e.g. exp, log, Bose factor correction)
  - and binary operations (+ ,-, /, *, ^),
  - function evaluation and
  - S(Q,w) model evaluation

- Work in parallel across multiple computers (i.e. distributed computing) on IDAaaS (ISIS Data Analysis as a Service), but also run on standalone workstations and laptops.

- Available for both Matlab and Python user environments.

- Creation of, and integration with, the application Euphonic to compute scattering from phonons for single crystals and powders from many third party codes (e.g. CASTEP, VASP, ABINIT, Quantum ESPRESSO, WIEN2k, CRYSTAL).

- Interface to SpinW for the calculation of scattering from spin waves.

UKRI Science and Technology Facilities Council ISIS

*PACE*

# PACE Features (2/2)

- A well-defined interface to other third party models code for S(**Q**,$\omega$) and S(|**Q**|,$\omega$), or user-written models written in Python or Matlab, so that these models can be straightforwardly used with the rest of the PACE functionality
- Proper account of instrument resolution function – for both single crystal and powder data – as a matter of routine.
- Fitting of parameters in modelling codes, with resolution function convolution
- Command line operation from Matlab or Python by default – but, if project resources permit, the construction of a GUI 'workbench' for managing the analysis of data with refinement of parameters in resolution broadened models for scattering

# PACE Project

**PACE project aims to deliver its goals by 31 December 2022**

- Refactoring of Horace is taking place in the background and new functionality is continually being introduced
- Minimum formal PACE release for beta testing with users at end of 2021/early 2022
- Iterative refinement and enhancements with users throughout 2022

**PACE is split into three major sub-projects**

1. Development of core framework for parallel and distributed computing architecture

2. Interface to third party simulation codes
    1. Phonon calculations (Euphonic)
    2. SpinW (now a separate project)
    3. Application Programming Interface (API) for generic user (Matlab, Python and C++), and 3rd party codes

3. Resolution convolution algorithms and $S(\mathbf{Q},\omega)$ model parameter optimisation against data

# PACE: Core data analysis framework

PACE will build on the capabilities of the current Horace software for single crystal data analysis, and extend to powders

Current Horace users will seamlessly see additional capabilities appearing – PACE is backwards compatible

- Functionality of Horace
  - Cut: 4D $\Rightarrow$ 3D, 2D, 1D from intermediate file
  - Cut: from any higher to any lower dimension object
  - Save/read to & from files
  - Symmetrisation (up to an order of magnitude counts increase)
  - Plot: 3D, 2D, 1D
  - Unary operations (exp, sin, tan, ...), detailed balance
  - binary operations (+,-,*,/,^)
  - Dimension expansion: from any lower to any higher dimension
  - User function evaluation
  - Simulation of S(**Q**,E) models, dispersion relations, and overplot dispersion relations
  - Fitting of S(**Q**,E) models

➔ Can build powerful functions out of these lower level functions

➔ Bespoke analysis for an experiment can be scripted

# PACE: Core data analysis framework

- Can run from Matlab or Python
  - Matlab operation will require a Matlab licence
  - Python operation will ***not*** require Matlab license (compiled Matlab code is hidden and freely distributable)

  ➔ Can use PACE as a library with other open source applications e.g. machine learning

- Functions will work on data sets too big to hold in memory (operations are file-backed)

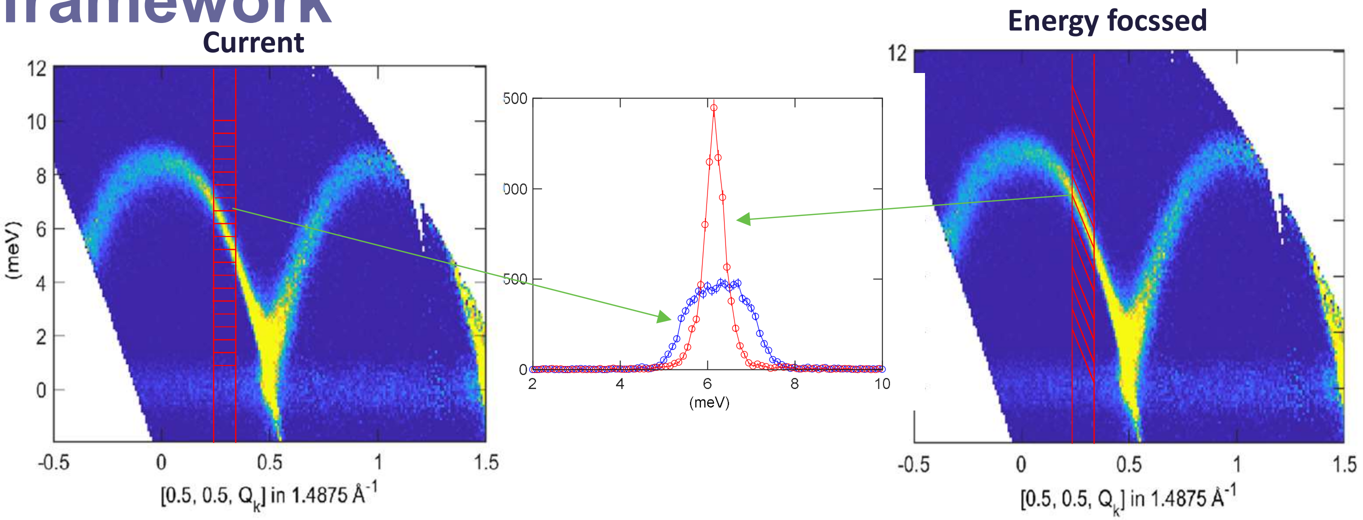
  ➔ Can run on powerful laptops (if sufficient disk space)

- PACE functions will work on multi-core and distributed computing
  - Takes advantage of large-scale computing if available
  - Compute-intensive operations on large data volumes become feasible
    - e.g. scattering function evaluation, resolution convolution, least-squares fitting

  Virtual clusters are being implemented on IDAaaS as part of the PACE project
  ➔ IDAaaS and the STFC cloud can be used with users doing all their analysis from IDAaaS

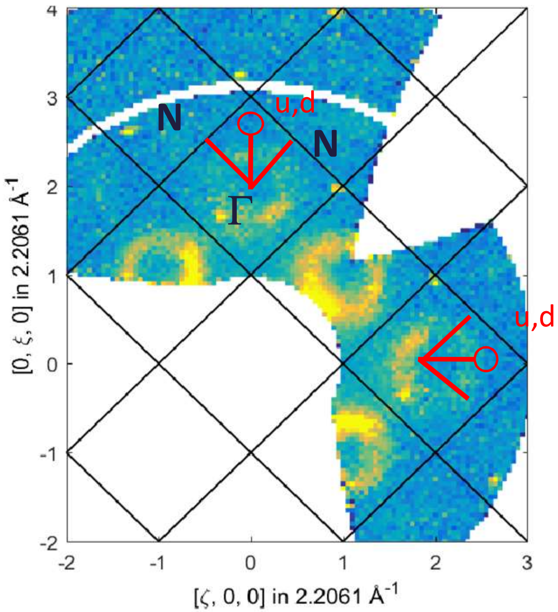
# PACE: Core data analysis framework

- Generalised cuts will be implemented
  - Arbitrary planes in reciprocal space
  - Powder averaging, cylindrical averaging
  - General user-provided transformation
  - Energy focused cuts *(already prototyped)*




- Symmetrisation on the fly
  - Specify symmetry operations as you take a cut
    - Translation, rotation, reflection
  - No need to symmetrise the .sqw file beforehand

  *(partial implementation already)*




8 equivalent cuts along [110]-type directions
- four in plane
- two diagonally out-of-plane up
- two diagonally out-of-plane down

Science and Technology Facilities Council

ISIS

*PACE*

# PACE: Core data analysis framework

**Current status:**

- Most functions have mow been rewritten so that they can operate on datasets too large to fit into memory
- A framework has been developed to support the parallelization of most operations, and tested alongside commands to create and destroy virtual compute clusters for running the operations
- The prototype of the Python command interface to PACE has been written, and used with the compiled Matlab core of PACE
- Professionalisation of software development
    - Important for the robustness of a critical multi-developer project
    - Continuous automatic verification of a large library of tests
    - Creation of builds for Windows and Linux operating systems

**Work to do:**

- Parallelisation of operations using the framework mentioned above has yet to be started.
- Production version of virtual clusters framework just started to be implemented
- Generalisation of cuts from data to include powder data, cylindrical averaging and other more general cuts is underway, but not completed *(requires major refactoring of core Horace functionality)*

# PACE: Scattering function evaluation

Simulating and fitting models for S(**Q**,ω) is a central requirement for PACE

- Simple user-written functions (in Matlab, Python, or C++)
- Euphonic – to compute phonon eigenvectors and eigenvalues from force constants (written as part of PACE)

  *(Almost ready for first public release – see below)*

- Interface to SpinW (linear spin wave theory)
  - Parallelisation of SpinW is a separate project funded by ISIS

  *(currently on hold until end of 2021, but will be finished by end of 2022)*

- Provision of a well-defined interface (Matlab, Python or C++)
  - PACE can interface to any program that is given a light software wrapper to be compatible with that interface

  *(Interface prototype has been tested)*

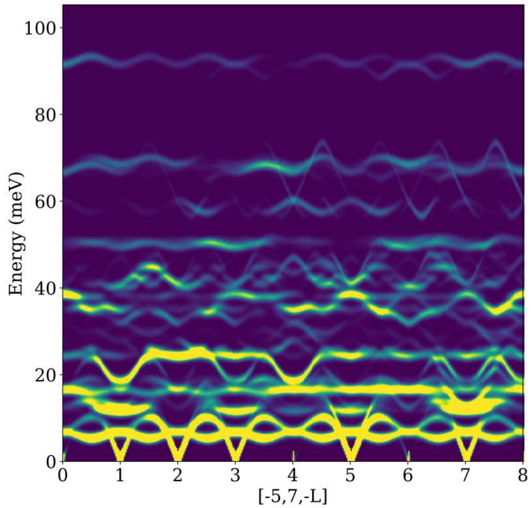
Scattering function evaluation is fully integrated into resolution convolution and parameter fitting

- This is a separate project within PACE – see later

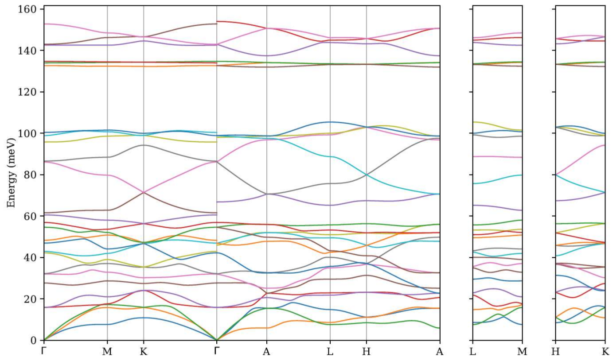
# PACE: Euphonic

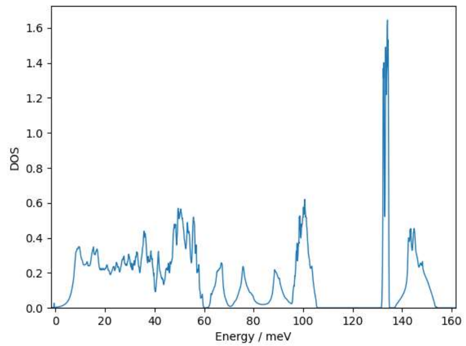
- Euphonic is a Python library to allow efficient simulation of phonons directly from force constants
- Currently supports CASTEP and Phonopy force constants
  ➔ Reads from VASP, ABINIT, Quantum ESPRESSO, WIEN2k, CRYSTAL…
- Core calculation is parallel C for speed

Simulation of Coherent Crystal Inelastic Neutron Scattering – $La_2Zr_2O_7$

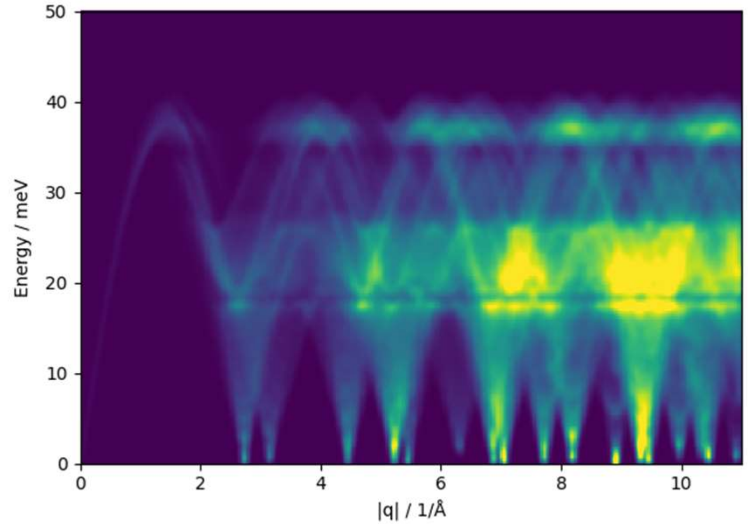


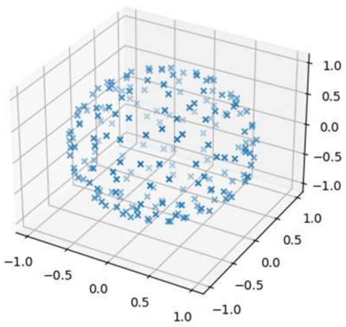

Other useful features – many also accessible via command-line tools

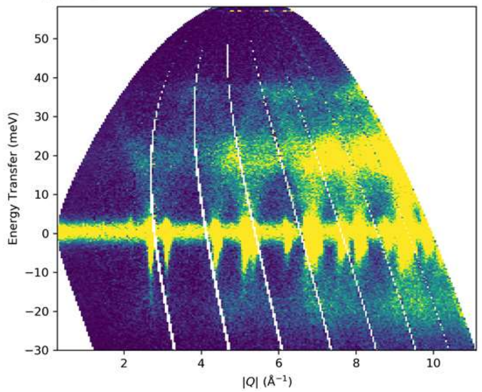



Coherent powder INS – Aluminium

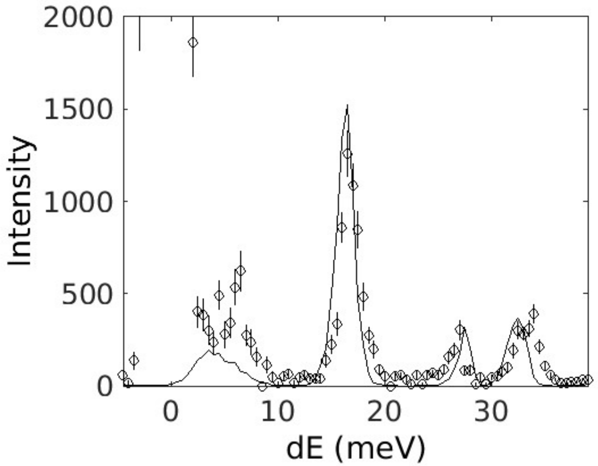


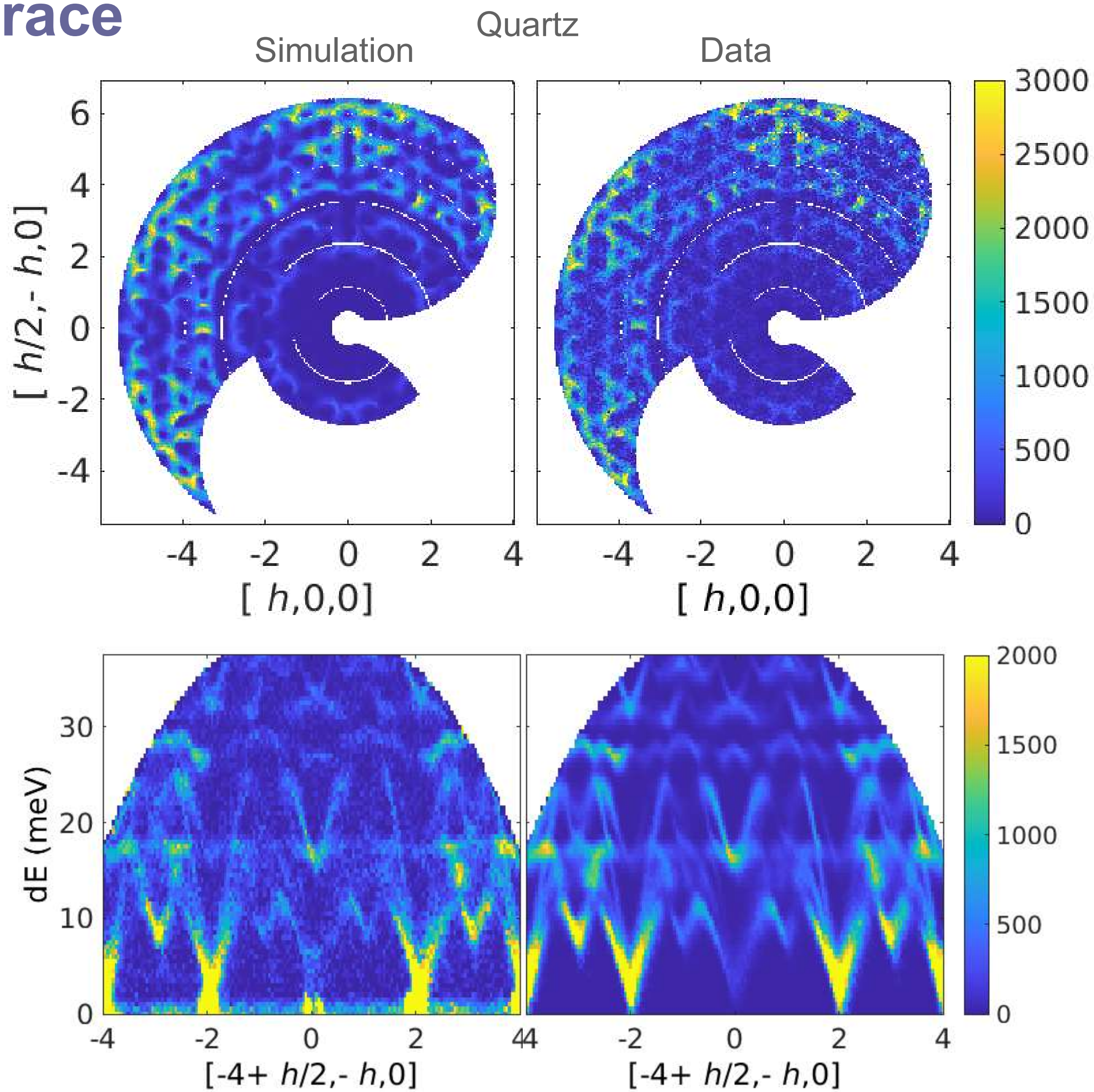





# PACE: Euphonic use with Horace

- Euphonic can be used to simulate crystal data in Horace using the Horace-Euphonic-Interface MATLAB add-on – allows direct comparison to experimental data
- Can be used with Tobyfit for resolution convolution



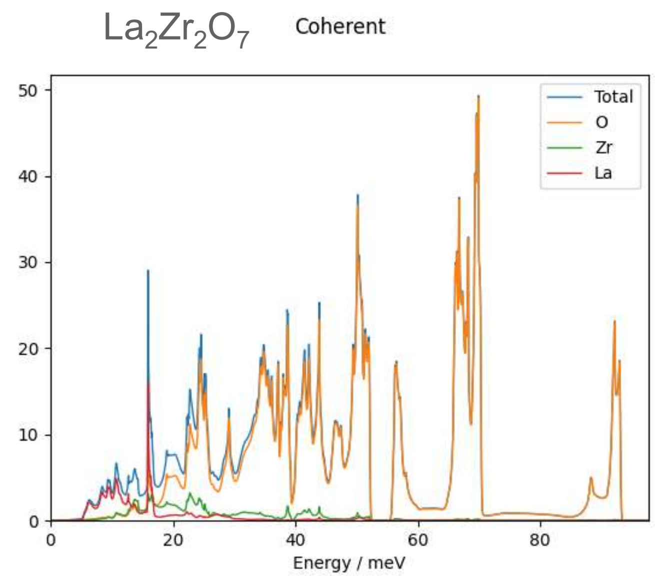

Science and Technology Facilities Council

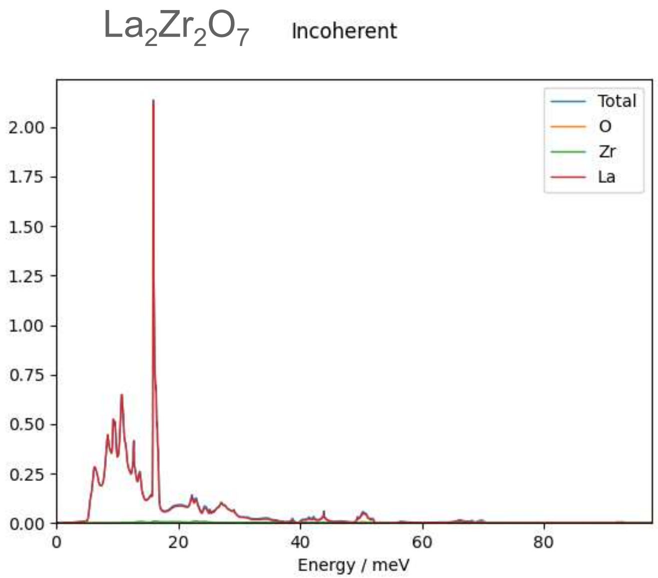
ISIS

# PACE: Euphonic

**Current status:**
- Library almost ready for Version 1.0.0 public release, with documentation, validation against existing codes (CASTEP phonon tools, Oclimax, Abs2ts)
- Horace-Euphonic interface complete

**Forthcoming:**
- Neutron-weighted phonon density of states (PDOS) – coming soon!
- Further development of powder simulations, including use with PACE
- Formalised benchmarking of performance for speed optimisation
- Integrated use of *Brille* library (also written for PACE) for faster interpolation



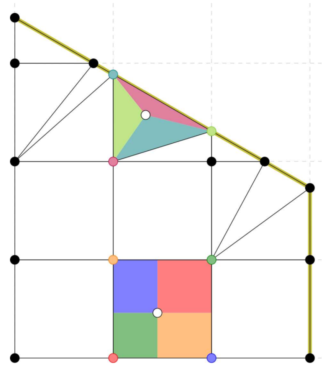



# PACE: *Brille* - symmetry aware interpolation *(for aficionados)*

**Goal:**

- Constructs Brillouin zone and irreducible Brillouin zone polyhedra for any space group
- Provides fast interpolation on a grid in irreducible Brillouin zone for any scalar, vector or tensor quantity

`brille` implements a **hybrid grid/mesh**:

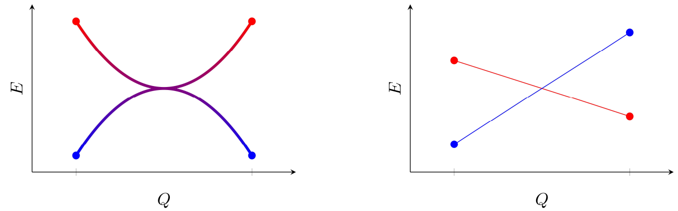


- ➢ Uses regular grid where possible
- ➢ Boundary cells use n-simplex mesh

**Features:**

- Reduces up-front expensive calculations of scattering models by computing on a grid in irreducible Brillouin zone
  ➔ makes fitting with resolution convolution on large datasets feasible

- Eigenvector character determination circumvents the interpolation problem across high symmetry directions/planes




eigenvector mixing prevents identification

distinct eigenvectors help identification

- C++/OpenMP with Python interface
  - Can drive modelling codes
  - Can be driven by modelling codes

- Library almost ready for public release, with documentation and validation

# PACE: Resolution convolution and parameter fitting

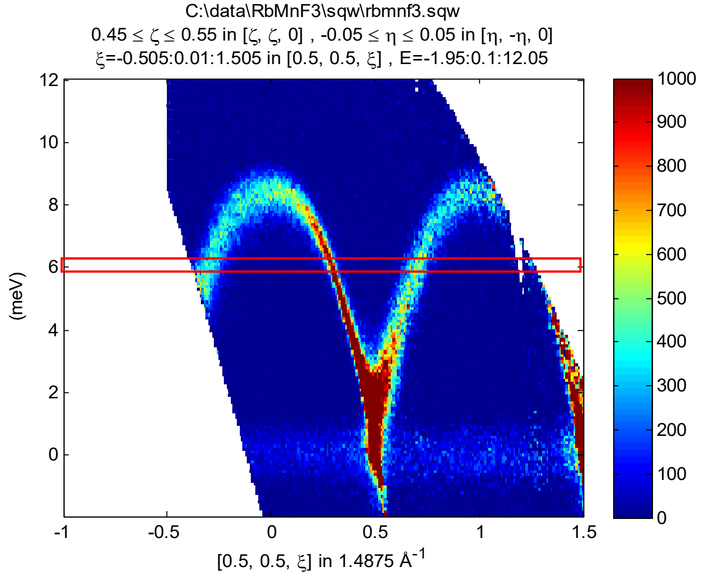
PACE will have resolution function convolution built in as default when simulating or fitting data

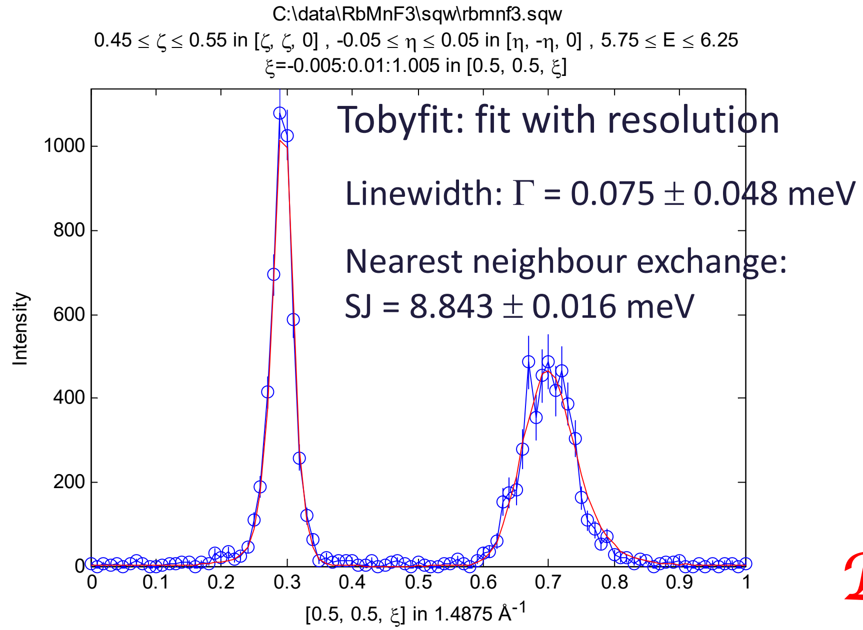
- Resolution can be very important in data interpretation:
  - Determination of weaker exchange constants
  - Linewidths of phonons or spin waves
  - Correlation lengths and correlation times of diffuse scattering
- Aspiration for PACE: 'Rietveld refinement for inelastic'

## Why resolution matters

The two peaks in the constant energy cut are symmetry equivalent – but resolution destroys that symmetry

Tobyfit fully accounts for the asymmetry

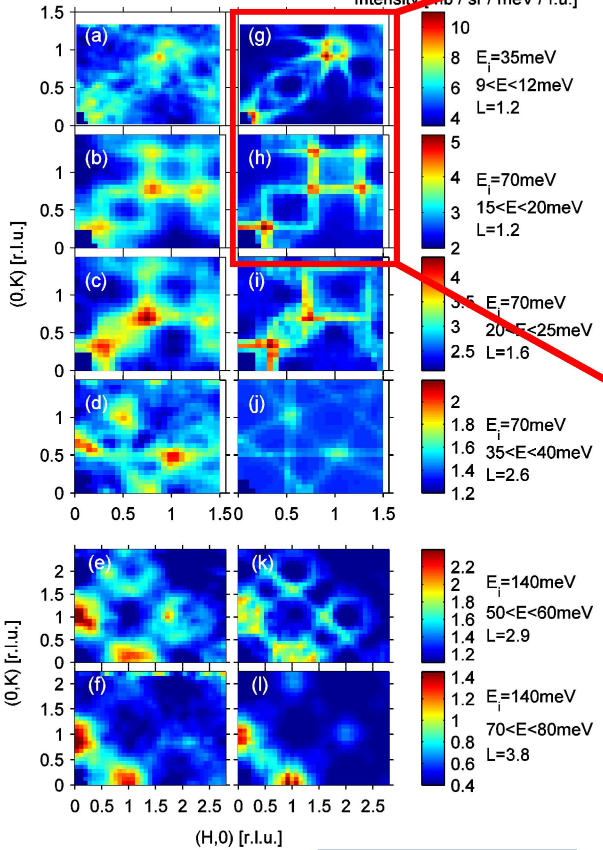


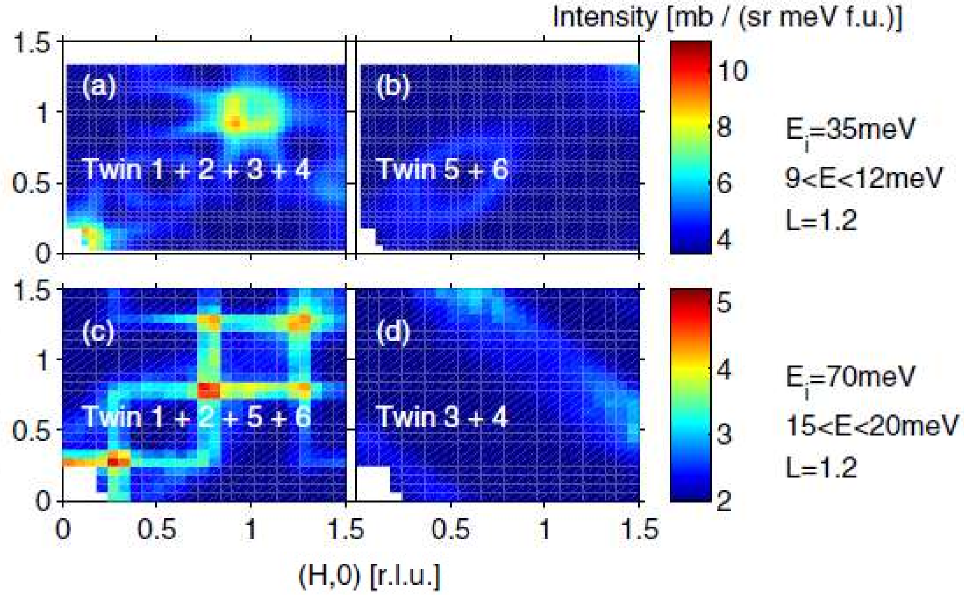




*PACE*

# Rietveld for inelastic: it really is feasible

## Example: $Pr_{0.5}Ca_{0.5}MnO_3$




- 8 magnetic atoms per cell; two species
- 6 magnetic exchange constants
- 6 twins
- 28 one-D cuts + 12 two-D slices

➔ Unambiguous solution of the exchange constants

R.Ewings, TGP et al PRB **94** 014405 (2016)

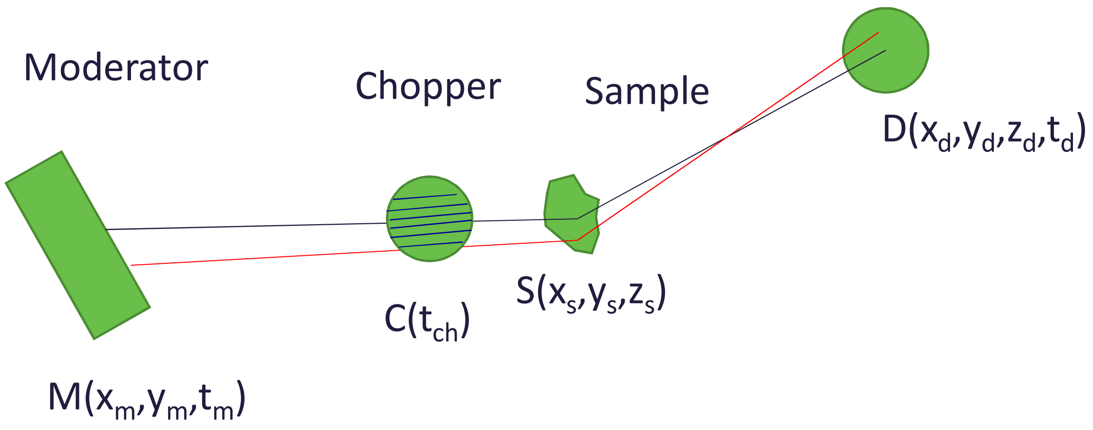
Science and Technology Facilities Council

ISIS

# Resolution convolution method

**Current:**
Tobyfit: randomly samples from instrument components and sample

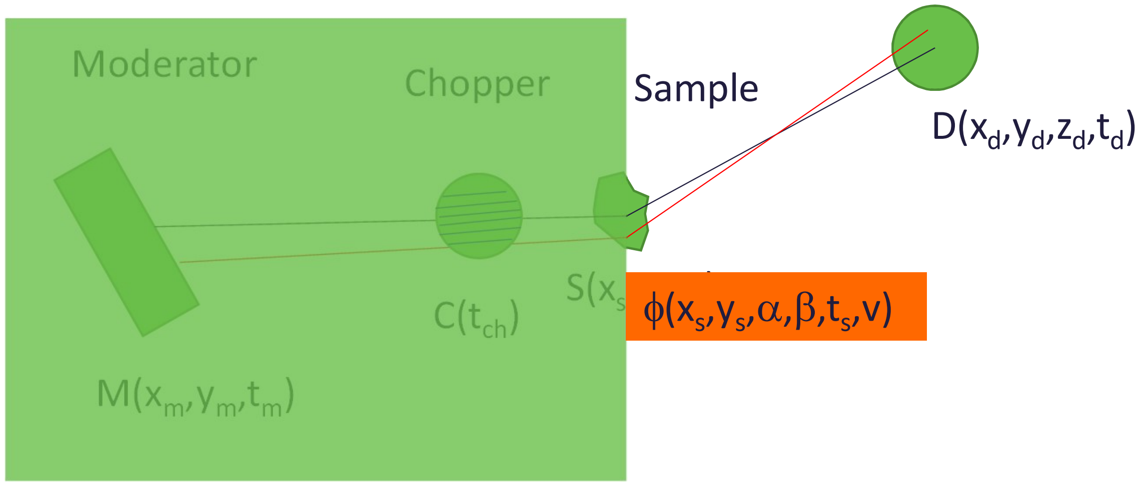
Approximates guide by 'effective moderator size' determined from McStas simulation




**Enhanced method to be implemented in PACE:**
Use full description in McStas to generate a table of neutron events at the sample position

Launch McStas job to create the table when start a run on instrument

Ready when user wants to perform simulation or fit




Science and Technology Facilities Council

ISIS

*PACE*

# PACE: Resolution

**Current status:**

- 'Tobyfit' method (semi-analytic resolution function with Monte Carlo integration) already implemented for single processor
- Refactored so that it is straightforward for PACE developers to implement additional instrument types and resolution models
- Features:
  - Refines parameters on arbitrary number of 1,2,3,4D data sets
  - Global or local S(**Q**,$\omega$) with global or local backgrounds: full control of modelling
  - Refines crystal alignment and moderator pulse width
  - Ability to bind parameters together in fit (to vary in fixed ratio)

Example: - spin wave dispersion: allow exchange parameters, intensity and linewidths to vary independently
- Bind exchange parameters so the dispersion relation is the same for all cuts
- But have independent intensities and widths for each cut

**Work to do:**

- Refactor fitting code to work in parallel – multi-core computers and distributed computing (computing clusters)
- Implement hybrid McStas – semi-analytical resolution function model for chopper spectrometers
  - It will properly account for neutron guide transport to sample
  - Cylindrical 3He detectors already properly accounted for
- GUI work-bench for managing fitting and resolution convolution