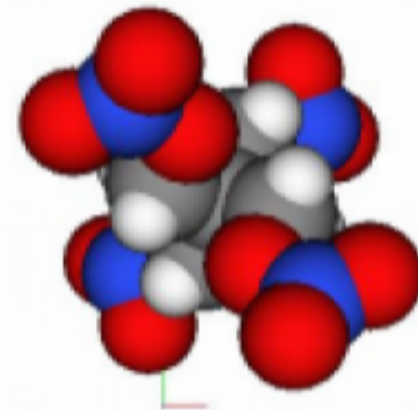


WHAT'S NEW WITH GSAS-II?

GSAS-2



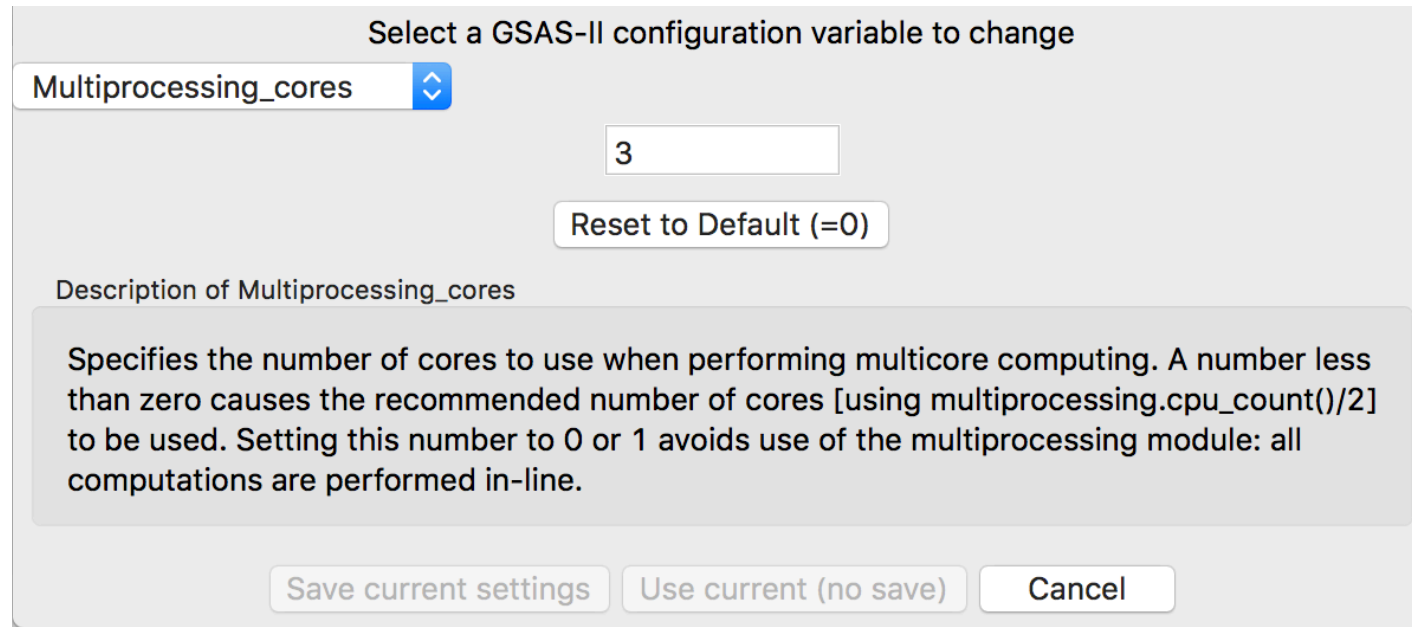
BRIAN H. TOBY

ROBERT B. VON DREELE

CXS
XST
XSD
APS
ANL
DOE
USA

PARALLEL PROCESSING DURING RIETVELD REFINEMENTS

- GSAS-II can optionally use multiple cores when available
 - Implemented for powder histograms only
 - Parallelizes across reflection loops: only useful when there are significant numbers of reflections
- To use, configuration option Multiprocessing_cores (preferences menu)



Select a GSAS-II configuration variable to change

Multiprocessing_cores

Reset to Default (=0)

Description of Multiprocessing_cores

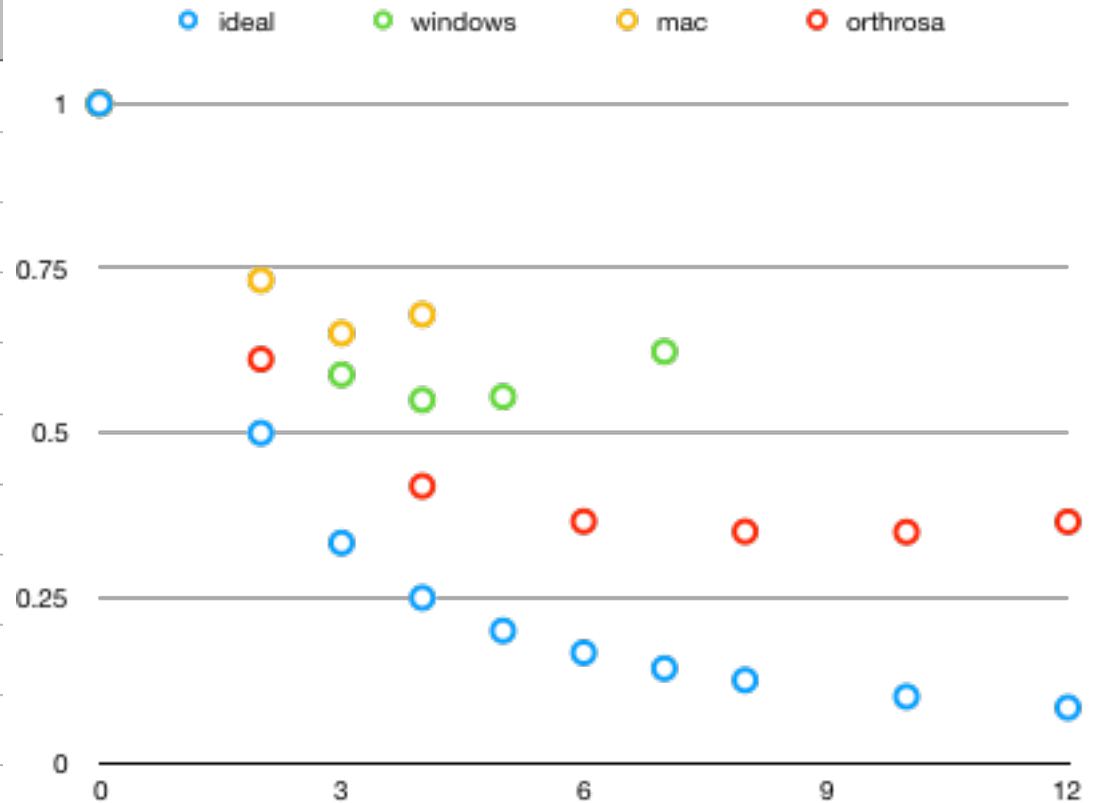
Specifies the number of cores to use when performing multicore computing. A number less than zero causes the recommended number of cores [using multiprocessing.cpu_count()/2] to be used. Setting this number to 0 or 1 avoids use of the multiprocessing module: all computations are performed in-line.

Save current settings Use current (no save) Cancel

MULTICORE PERFORMANCE

For computationally demanding refinements, the optimal seems to be ~ number of real cores (50% of virtual)

T3R3/ cycle (sec)	windows	mac	orthrosa
0	59.11	57.79	79.78
2		42.3	48.82
3	34.79	37.64	
4	32.52	39.28	33.46
5	32.81		
6			29.19
7	36.85		
8			27.95
10			27.90
12			29.15



CONSTRAINTS IMPROVEMENTS

- The Constraints GUI now checks for conflicts in user-defined constraints with symmetry-generated constraints
- Symmetry generated constraints can now be displayed (separate tab)
- Error messages provide better detail
- The GUI tries to prevent creating constraints that are invalid
- As before, invalid constraints will prevent refinement (also with improved error messages)

If you are able to generate an invalid constraint, please provide that as an example to Brian.

SCRIPTING INTERFACE TO GSAS-II

Work of Jackson O'Donnell (U of C student w/Maria Chan @ CNM)

```
6 import sys
7 import GSASIIscriptable as G2sc
8
9
10 datafile = sys.argv[1]
11 database, dataname = os.path.split(datafile)
12 if len(sys.argv) > 2:
13     projname = sys.argv[2]
14 else:
15     projname = os.path.join('/home/odonnell/amno2_data',
16                             dataname.replace('.raw', '.proj'))
17
18 instprms = '/home/odonnell/amno2_data/instprm.txt'
19 phasefile = '/home/odonnell/amno2_data/phase.txt'
20 refinementsfile = '/home/odonnell/amno2_data/refinement.txt'
21
22 proj = G2sc.G2Project(filename=projname)
23 hist = proj.add_powder_histogram(datafile, instprms)
24 phase = proj.add_phase(phasefile, histograms=[hist])
```

Line	
1	{
2	"refinements": [{
3	"set": {
4	"Limits": [0.7, 7.8],
5	"Background": {
6	"no. coeffs": 6, "refine": true
7	}
8	}
9	},
10	{
11	"once": { "LeBail": true },
12	"set": { "Cell": true }
13	},
14	{
15	"set": { "Sample Parameters": ["DisplaceX"] }
16	},
17	{
18	"set": { "Instrument Parameters": ["U", "V", "W", "X", "Y"] }
19	},
20	{
21	"set": { "Atoms": { "Mn": "X" } }
22	},
23	{
24	"set": { "Atoms": { "O1": "XU",
25	"O2": "XU",
26	"Mn": "XU" } }
27	},
28	{
29	"set": { "Atoms": { "O5": "F" } }
30	}]
31	}