JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ

Getting started with JaBS

Introduction and usage instructions for JaBS 0.7.5 (released Nov 11, 2024)

Jaakko Julin jaakko.julin@jyu.fi

This document was last modified 12.11.2024

Before we start...

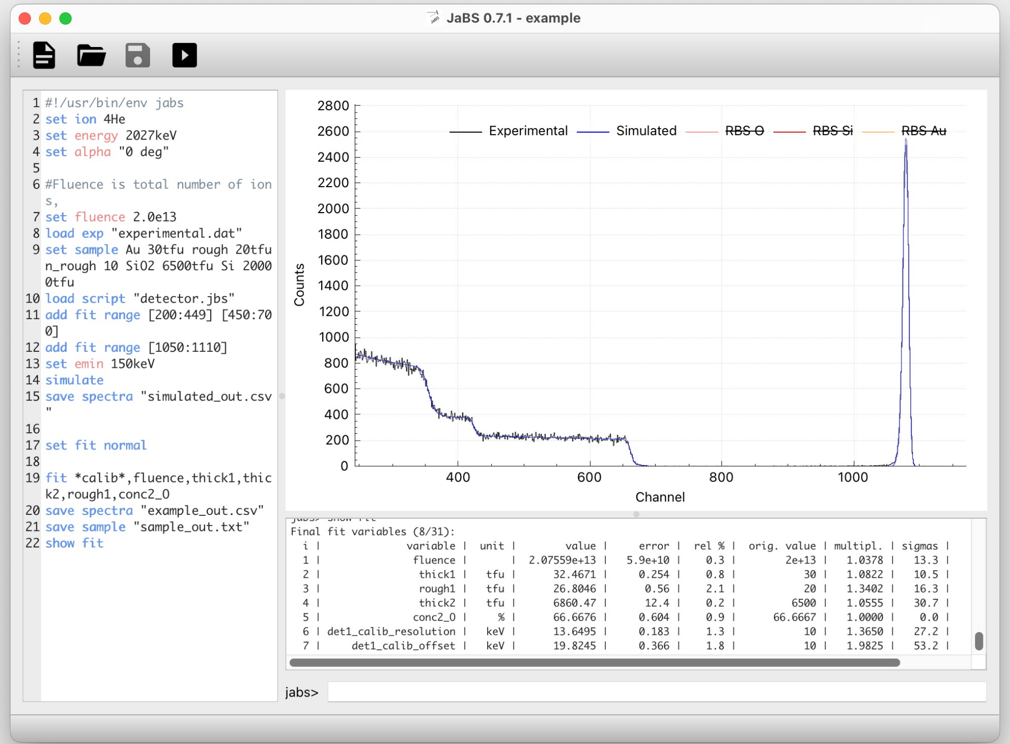
- Caveats of this document
  - This is work in progress
  - Instructions reflect the version given on the title slide, but may include some features that are only present in the latest version from git version control (https://github.com/JYU-IBA/jabs)
  - known issues in latest binary distributions are at the end of this document, fixed bugs are in release notes
- Latest git version README in GitHub, check out latest releases too
- JaBS has not been published yet in any scientific publication
  - Each release is archived on Zenodo with a unique DOI
  - You can cite “all versions” using DOI 10.5281/zenodo.6362701 but please also include the version in the citation, since the DOI resolves to latest version in Zenodo

Jaakko’s Backscattering Simulator (JaBS)

DOI 10.5281/zenodo.6362701

- Simulates RBS, EBS, NRA and ERD spectra
  - Strictly limited to particle-particle interactions currently
- Open source (GPLv2 or newer) license
  - GitHub, Zenodo
- Uses JIBAL library
  - Stopping
  - Isotopes, elements (abundances)
- Multi-platform
  - Windows
  - Linux
  - macOS

https://github.com/JYU-IBA/jabs



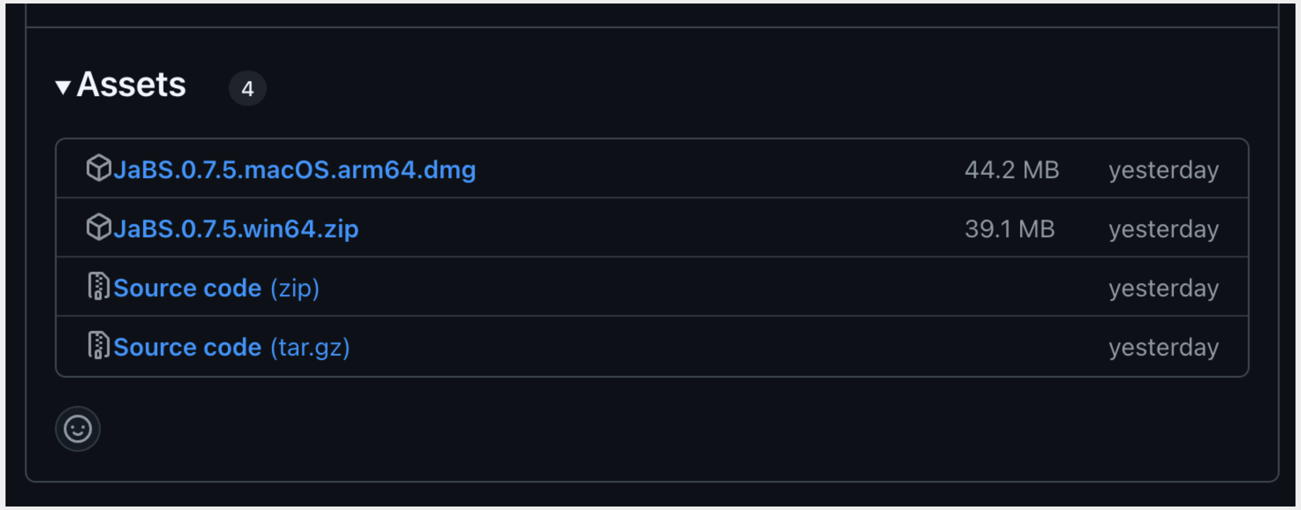
```
#!/usr/bin/env jabs
set ion 4He
set energy 2027keV
set alpha "0 deg"

#Fluence is total number of ions,
set fluence 2.0e13
load exp "experimental.dat"
set sample Au 30tfu rough 20tfu n_rough 10 SiO2 6500tfu Si 20000tfu
load script "detector.jbs"
add fit range [200:449] [450:700]
add fit range [1050:1110]
set emin 150keV
simulate
save spectra "simulated_out.csv"

set fit normal

fit *calib*,fluence,thick1,thick2,rough1,conc2_O
save spectra "example_out.csv"
save sample "sample_out.txt"
show fit
```

```
Final fit variables (8/31):
  i |              variable |  unit |        value |     error |  rel % |  orig. value | multipl. | sigmas |
  1 |               fluence |       | 2.07559e+13 |   5.9e+10 |    0.3 |        2e+13 |   1.0378 |   13.3 |
  2 |                thick1 |   tfu |     32.4671 |     0.254 |    0.8 |           30 |   1.0822 |   10.5 |
  3 |                rough1 |   tfu |     26.8046 |      0.56 |    2.1 |           20 |   1.3402 |   16.3 |
  4 |                thick2 |   tfu |     6860.47 |      12.4 |    0.2 |         6500 |   1.0555 |   30.7 |
  5 |               conc2_O |     % |     66.6676 |     0.604 |    0.9 |      66.6667 |   1.0000 |    0.0 |
  6 | det1_calib_resolution |   keV |     13.6495 |     0.183 |    1.3 |           10 |   1.3650 |   27.2 |
  7 |     det1_calib_offset |   keV |     19.8245 |     0.366 |    1.8 |           10 |   1.9825 |   53.2 |
```

jabs>

GitHub releases

- See this page https://github.com/JYU-IBA/jabs/releases/
- See “Assets”
  - You can find ready-to-use packages of JaBS there for 64-bit Windows (…win64.zip) and macOS (…dmg) running on Apple Silicon (M1 – M4, …?) processors
  - Stopping data etc are included

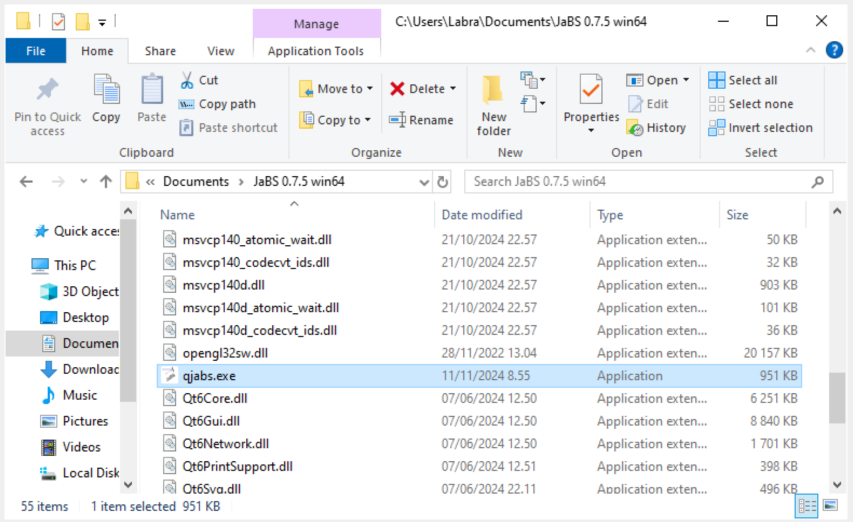


▾ Assets 4

JaBS.0.7.5.macOS.arm64.dmg	44.2 MB	yesterday
JaBS.0.7.5.win64.zip	39.1 MB	yesterday
Source code (zip)		yesterday
Source code (tar.gz)		yesterday

Running JaBS

- Windows
  - Extract files from the zip
  - Run *qjabs.exe* for JaBS with Qt graphical user interface, *jabs.exe* for the command line version
  - Hint: qjabs.exe can be associated with *.jbs files, making opening them easier
- macOS
  - Open DMG, click on *JaBS.app* or drag it to *Applications*
  - When running the app, you will get an error since the app has not been signed by a known developer. See *Privacy & Security* settings to "*Open Anyway*"

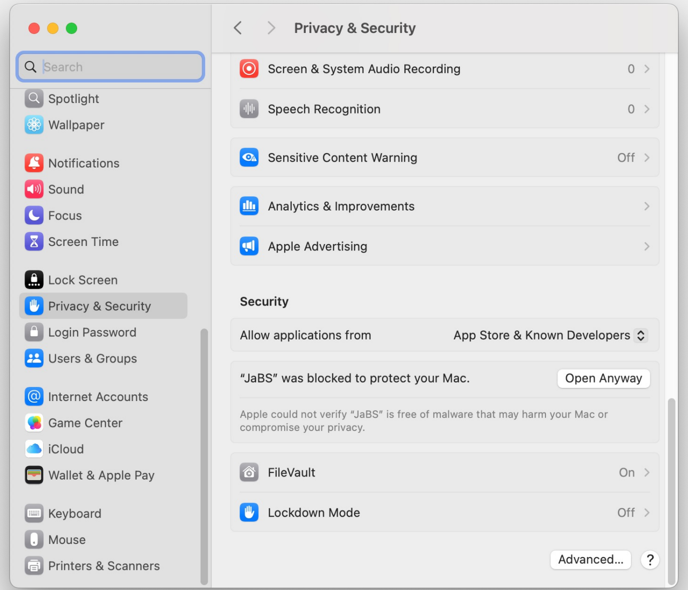


C:\Users\Labra\Documents\JaBS 0.7.5 win64

« Documents › JaBS 0.7.5 win64

Name	Date modified	Type	Size
msvcp140_atomic_wait.dll	21/10/2024 22.57	Application exten...	50 KB
msvcp140_codecvt_ids.dll	21/10/2024 22.57	Application exten...	32 KB
msvcp140d.dll	21/10/2024 22.57	Application exten...	903 KB
msvcp140d_atomic_wait.dll	21/10/2024 22.57	Application exten...	101 KB
msvcp140d_codecvt_ids.dll	21/10/2024 22.57	Application exten...	36 KB
opengl32sw.dll	28/11/2022 13.04	Application exten...	20 157 KB
qjabs.exe	11/11/2024 8.55	Application	951 KB
Qt6Core.dll	07/06/2024 12.50	Application exten...	6 251 KB
Qt6Gui.dll	07/06/2024 12.50	Application exten...	8 840 KB
Qt6Network.dll	07/06/2024 12.50	Application exten...	1 701 KB
Qt6PrintSupport.dll	07/06/2024 12.51	Application exten...	398 KB
Qt6Svg.dll	07/06/2024 22.11	Application exten...	496 KB

55 items 1 item selected 951 KB



Privacy & Security

Screen & System Audio Recording	0
Speech Recognition	0
Sensitive Content Warning	Off
Analytics & Improvements	
Apple Advertising	

Security

Allow applications from	App Store & Known Developers

"JaBS" was blocked to protect your Mac.

Apple could not verify "JaBS" is free of malware that may harm your Mac or compromise your privacy.

FileVault	On
Lockdown Mode	Off

Jaakko’s Backscattering Simulator (JaBS)

- Brief timeline
  - Feb 2021: Development start (first commit), first spectra simulated in less than a week
  - April 2021: Used for *real work*
  - August 2021: GUI (using Qt)
  - October 2021: Introduced at IBA 2021 (virtual) as an RBS code, first distributed version (0.4.8)
  - March 2022: first so-called release in GitHub, archived in Zenodo
  - October 2023: Re-introduced at IBA 2023 (Toyama, Japan) as an RBS, EBS and NRA code, version 0.7.3

Jaakko's Backscattering Simulator (JaBS)

- April 2022: version 0.6.0, releases are tested and release notes are made, usability improvements, faster fitting
- May 2022: NRA support, "accurate" mode
- July 2022: ECAART poster presentation as an RBS, EBS, NRA, ERD code with multidetector support
- Later 2022, early 2023: Bugs have been crushed, import of IDF, support for transmission geometry, arbitrary roughness, dual scattering improved, simulation routine rewritten, multiprocessing…
- September 2023: IDF export
- October 2023: IBA&PIXE2023 presentation
- Entirety of 2024: slow progress, but bugs have been fixed. Making new releases has been made easier for the developer.

What’s new? What’s old? Why should I care? 🤨

- Multidetector support
  - Single sample, single beam
  - Somewhere between SIMNRA and NDF
- Scriptable
  - Not programmable!
  - Allows “code reuse”
    - rapid analysis of similar samples, reference samples etc.
- Open source
  - Could be turned into a library and used by other programs as a simulation engine
  - Not a black box (if you can read C 😄)!
- Nice fitting
  - Finds best local fit and gives error bars
- Practical
  - Not too difficult to learn and use
  - Fast to define samples etc
  - Command line executable makes it easy to integrate in analysis workflows
- Accurate
  - Not suitable for HI-ERDA
  - Similar accuracy to SIMNRA in most cases (depends on user)
- Tested and benchmarked
  - Bugs are always possible, but bundled tests should catch major issues
- Available on all major platforms

Programming things

- Native Windows (10+), Linux and macOS support
- Programmed in C / GUI in C++
  - Semi-modern, C99, CMake build system
  - Profiled, memory leaks checked
  - Multi-threading using OpenMP
  - Scriptable, but user readable command language
  - Learning by example
  - Import of IDF files (incomplete, but SIMNRA exported files should mostly work)
  - Planned: wizards (answer questions to make a script)
- Qt 6 GUI (QJaBS, *qjabs.exe*) or command line (*jabs.exe*)
  - Same codebase
- Approximately 16000 lines (as of Mar 2023) of code in .c/.h files, not including Qt GUI

Basics of simulation (how JaBS works)

- Initialize
  - Calculate angles, some constants etc
- Incident ion
  - Tabulate depth vs energy vs straggling
  - Strategy is to pick an energy loss ΔE (user input or "automatic"), evaluate stopping S, take depth step $x = \Delta E / S$
  - Calculate real energy loss by solving differential equation with fourth-order Runge-Kutta
  - Stop at boundaries of layers
  - Calculation does **not** depend on knowledge of detector properties etc
- Brick
  - For some E_0 calculate energy at detector (E_{det}), then for lower energy $E_0 - \Delta E$, the "space" between two intervals is called a brick.
  - Required ΔE estimated based on detector resolution and estimating dE_{det}/dE_0
  - Concentration of elements may change throughout the bricks in general case
  - **Number of counts in a brick** determined in the worst case by taking account energy change in the brick, width of energy distribution (straggling) and changing concentrations!
    - Multidimensional integral
    - …or just ignore straggling and use mean energy and mean concentration
- Convolute
  - Each brick produces a spectrum with a box convoluted by a Gaussian with different σ at either end
  - JaBS uses differences of an approximative gaussian CDF to do this convolution, the solution is novel, fast and accurate enough

JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ

Features of JaBS

What can it do?

JaBS can do two (2) things!

1. Simulate
   - Rutherford backscattering spectrometry (RBS)
     - Cross sections either pure Rutherford or Andersen (from JIBAL)
     - Kinematics with both solutions
   - Elastic recoil detection analysis (ERDA)
     - Cross sections like in RBS
   - Elastic backscattering spectrometry (EBS)
     - Cross sections from R33 files
     - Possible to combine R33 files with built-in RBS reactions for full energy range
   - Nuclear reaction analysis
     - Cross sections from R33 files
     - Kinematics calculated correctly (I hope!)

JaBS can do two things!

2. Fit
   - Many parameters are available as fit parameters
     - Which parameters are available depends on sample and detector description
     - Multiple parameters can be fitted at the same time
       - Even too many; no checks are made if there are more fit parameters than degrees of freedom, e.g. fitting concentrations of all elements in a layer does not make much sense
     - Parameters can be defined with wildcards, e.g. *calib* will match ALL calibration parameters of all detectors
   - Multidimensional non-linear least-squares (Levenberg–Marquardt)
     - Finds local best fit
       - User should make an initial guess by making a simulation, then changing ”*sim*” command to “*fit* …”
     - Fit can be far off, if number of fitted parameters is low ”robust system”
   - Multiple fits can be made in the same session
     - Unsuccessful fits return the fitted variables back to original values, successful fits will keep the fitted values

Tunable speed / accuracy

- Multiple profiles
  - accurate mode uses adaptive integration
  - **default** profile provides enough accuracy in most cases
  - **fast** profile
    - Mean concentration and energy (see previous slide)
    - Bigger depth steps
  - Two extra profiles: **brisk** and **improved**
- These parameters can also be changed individually

Fitting

- Multidimensional nonlinear least-squares fitting
  - Algorithm is Levenberg – Marquardt from GSL using QR decomposition, no geodesic acceleration is used. Jacobian is evaluated numerically using finite difference method.
  - Multidetector fit is no different than single detector fit. Since some parameters (like detector calibration slope) only affect the particular detector spectrum, the Jacobian will have zero values. JaBS knows not to calculate all detector spectra in this case.
- Calculates also fit errors and covariance matrix using the Jacobian
- Error calculations take into account renormalization in some cases (concentration fitting)
- Two phase fitting
  - Enable fast mode and fit, then run with user defined profile
  - First phase can be skipped
- Computation is parallelized
  - Evaluation of Jacobian requires simulation of one detector spectrum (detector specific parameter) or all detector spectra (others) **for each** fit parameter. These are done simulated in parallel with OpenMP dynamic scheduling.
  - For each fit iteration all spectra are simulated once
    - Parallel for each detector
  - Linear algebra and rest of L-M is **not** parallel (GSL compatibility)

Sample models

- Simulation uses internally linearly varying concentrations (like mcerd)
  - Layered model: add two points for each layer, constant concentration in between
  - The code will not waste time on interpolating concentrations when it knows the sample model was originally layered
- Not just elements but **materials** (c.f. molecules in NDF), e.g. SiO2, we'll come back to this
  - Possible to keep elemental ratios constant during a fit
- Materials are turned to isotopes for the actual simulation
  - Isotopes are combined (so they are unique)
- Ad-hoc corrections for each layer or depth interval are possible
  - Bragg correction (stopping)
  - Straggling
  - Yield
  - "Channeling" (yield of last layer, linear w.r.t. energy)

Detector related features

- Multiple detectors in arbitrary geometries, automatic exit angle calculation
  - Must supply *two* angles for each detector
    - Default φ is 0° (IBM) if nothing else is given, also 180° is IBM, while Cornell geometry is 90° and 270°
- Non-linear energy response
  - Polynomial calibration, arbitrarily high degree possible, but monotonicity is strictly enforced!
  - Multilayer foil in front of detector
- ToF detector
  - Energy spectra
- Different calibrations for different reaction products, e.g. p, He, …
  - This is element (Z) specific
  - Different isotopes always have the same calibration

Cross sections and reactions

- Built in cross sections (Andersen) and automatic RBS / ERD reactions
  - Alternative (–) solution for RBS (since late 2022)
- Nuclear reactions or non-Rutherford via R33 files
  - Files created by SigmaCalc work fine
- Arbitrary cross sections via plugin (dynamically loaded)
  - Not supported on Windows, since it uses dlsym()
- Can be limited to an energy interval based on incident energy
  - Combine multiple “reactions”

Straggling and related effects

- Electronic energy loss straggling via JIBAL
  - Bohr, Chu, Yang, …
- Geometric straggling (off by default),
  - Not limited to IBM or Cornell geometry
  - Both beam spot and finite detector size considered (correlated)
- Non-statistical straggling due to changes in stopping
- Roughness, or thickness variation, by simulation of multiple subspectra
  - Gamma roughness (like SIMNRA)
  - Arbitrary roughness, lateral structures, i.e. probability vs. thickness

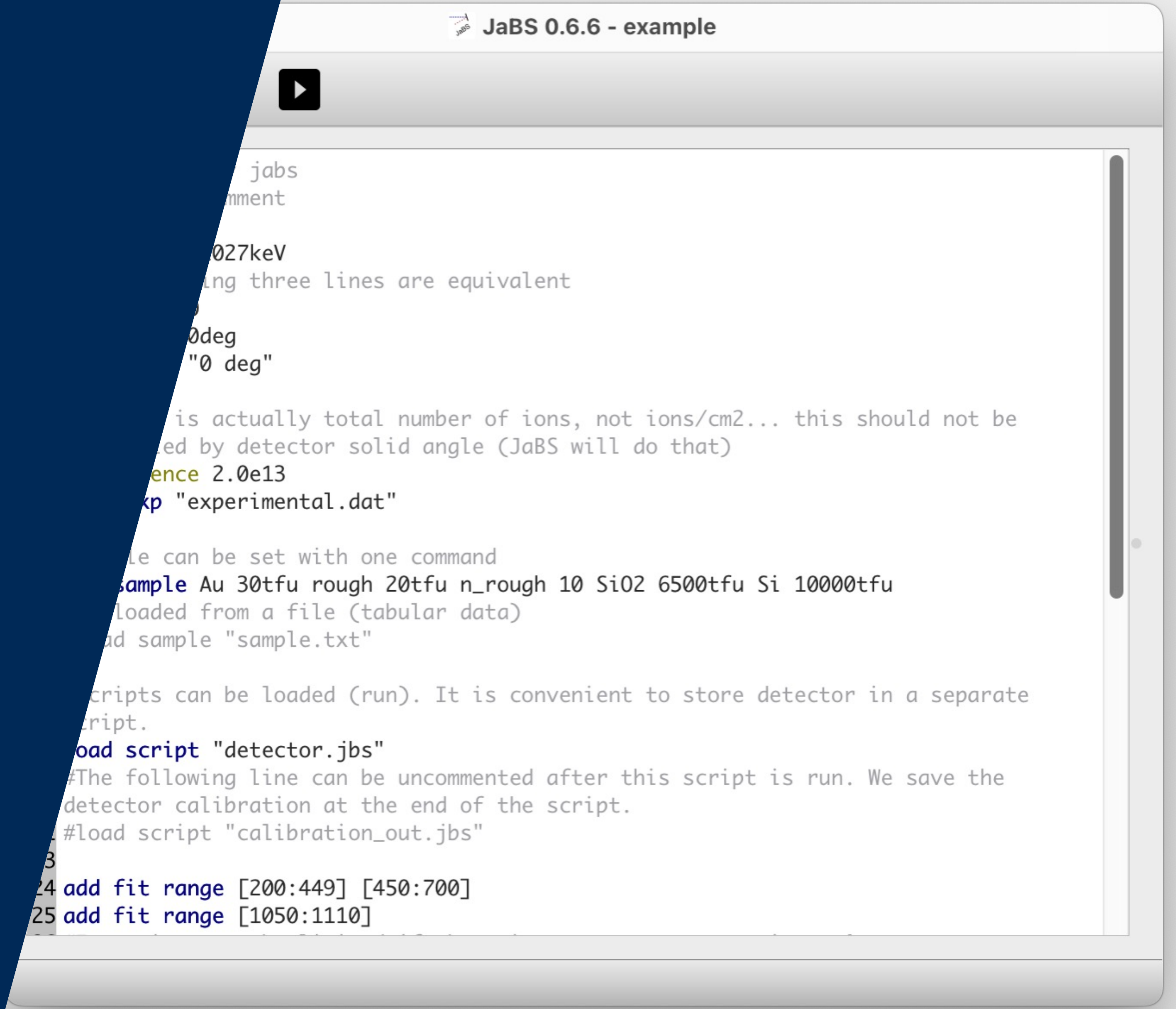
Multiple scattering

- Small angle multiple scattering models are not implemented (yet)
- Large angle plural scattering
  - Similar to dual scattering in SIMNRA
  - First scattering is always RBS (scatter – scatter, scatter – recoil) with Andersen cross sections
  - Compared to SIMNRA 7 takes into account more trajectories

JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ

Use of JaBS for routine RBS analysis 🤗

…or how to define you experiment, load data etc and how to perform simulations and fits



```
 jabs
mment

027keV
ing three lines are equivalent
)
0deg
"0 deg"

 is actually total number of ions, not ions/cm2... this should not be
ed by detector solid angle (JaBS will do that)
ence 2.0e13
xp "experimental.dat"

le can be set with one command
sample Au 30tfu rough 20tfu n_rough 10 SiO2 6500tfu Si 10000tfu
loaded from a file (tabular data)
ad sample "sample.txt"

cripts can be loaded (run). It is convenient to store detector in a separate
cript.
oad script "detector.jbs"
#The following line can be uncommented after this script is run. We save the
detector calibration at the end of the script.
#load script "calibration_out.jbs"

add fit range [200:449] [450:700]
add fit range [1050:1110]
```

JaBS scripts: not so difficult?

- Interactive use possible, recommended to write files
  - file extension *.jbs* can be associated with JaBS, on Linux desktops this is done automatically if JaBS is installed
- Qt GUI
- JaBS scripts are used to *set* / *add* / *load* / *reset* / *save* things
  - Most commands start with one of these words
- In general removing things is not implemented, due to programmer “laziness” (lack of time)
  - Scripts should be “additive”
- Not a programming language
  - No user definable variables
  - No flow control
  - No calling of routines with arguments
- User writable and hopefully readable
- Loading of scripts within scripts possible
  - Good practice to define experiment and detector in a separate file (or two) and load them using *load script*
  - ”One sample – one file” philosophy
- **Learn by example**
  - Lack of documentation
- **Import an existing simulation (IDF)**
  - Note that SIMNRA 7 xnra files *are* IDF and can be imported as-is *without* using the IDF export feature

Basic JIBAL concepts

- "Isotope"
  - *7Li*, *28Si*, *1H* are all valid, while *Li*, *Si*, *H*, *7li*, are not
  - Abbreviations for commonly used ions and for R33 file support:
    - ***p*** is equal to ***1H***, ***d*** is ***2H***, ***t*** is **3H**, ***a*** is **4He**, ***h*** is **3He**
  - Isotopes in JaBS are immutable (properties can not be changed) and unique (no two copies exist of the same isotope)
- "Element"
  - Contains an arbitrary number (≥1) of isotopes of the same element (same Z)
  - Are mutable
    - Arbitrary isotopic mixture may be defined
  - *Li*, *Si* and *H* are all valid
    - Natural isotopic mixture (average numbers from IUPAC) are assumed, isotopes with abundance below 1 ppm (atomic) are not included
  - *7Li*, *28Si* and *1H* are all valid too!
  - So is "*6Li0.075 7Li0.925 O2*"
    - The definition is the part inside the quotes, but quotes may need to be supplied since there is a space in the middle
    - Isotopic mixture is normalized to unity
- "Material"
  - One or more elements, with concentrations, e.g. **SiO2**, "*Si0.333333O0.666667*", "*28Si 16O2*"

Script syntax, notation, parsing

- Commands and parameters are separated by a space ( )
- Double quotes (“) must be used to give parameters with spaces in them, otherwise they are optional
  - *load script “C:\Users\Jaakko Julin\Documents\RBS\example.jbs”*
- One line, one command, but…
- …commands have subcommands, multiple subcommands may (usually) be given in a single line
  - *set detector resolution 20keV slope 2keV offset -20keV*
- Commands may be abbreviated as long as they are unique (*set det res 20keV*)
  - Qt GUI highlights commands
  - Avoid excessive abbreviating, since new commands may be introduced in new versions
- Hash/pound sign (#) starts a comment
  - Beginning of line or elsewhere (but not inside double quotes)

Working directory

- If filenames are used they are either **absolute** (like on previous slide) or relative to **current working directory**
  - This is how most programs (should) work
- In command line use the "*cwd*" is the directory in which *jabs* was started
  - Use jabs command *cwd* or *pwd* to see the current working directory
  - Use jabs command *cd* to change it
- In QJaBS current working directory is set to the directory of the script file (*.jbs)* when a script is loaded
- This strategy makes it easy to use relative file paths to refer to files in the same directory as the script
  - When using the command line, run jabs in the same directory as the script file

System of units

- Most physical parameters require a unit
- SI units are used internally, units are simply conversion factors to SI
- Space between numeric value and unit may be omitted
- SI prefixes and exponential notation are supported
- Decimal separator is **always** a dot (.), JaBS intentionally overrides language and regional settings
- The following are all equivalent:
    - *set energy 2.0MeV*
    - *set energy “2000 keV”*
    - *set energy 3.2043533e-13*

Typical structure of a script file

- Define beam
  - Good idea to define it first
  - Loading reactions uses the information
- Define sample
  - Loading reactions can also use this information
- Define reactions
  - Optional in basic use (RBS), see advanced use for details
- Define fit ranges if a fit is performed
- Simulate / Fit
- Save sample, spectra etc.
- Something else?
  - Fit changes current parameters so you are not limited to performing one simulation/fit

Defining your beam

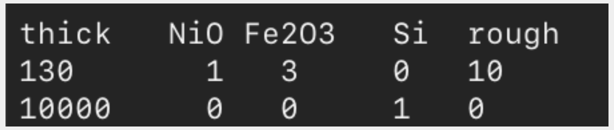
- Ion
  - *set ion 4He*
- Energy (default 2.0
  - *set energy 2MeV*
- Number of incident particles (incorrectly called *fluence)*
  - *set fluence 1.0e14*
- Energy broadening of beam (FWHM)
  - set energy_broad 0keV
- The values given above are the defaults for a new JaBS session (after *reset*), but it is good practice to define most of these always as defaults could change

Sample model

- No default sample, one must always be set
- Inline notation
  - Simple: *set sample Au 30tfu SiO2 6500tfu Si 20000tfu*
  - Complicated: *sample Au 30tfu rough 10tfu n_rough 30 density 19.3g/cm3 SiO2 6500tfu density 2.2g/cm3 Si 20000tfu*
  - Compounds will be split into elements (i.e. SiO2 becomes Si and O with appropriate concentrations)
  - Set *sample nosimplify* … can be used to override this behaviour, but note that concentration table becomes an identity matrix
- Tabular data
  - Sample model can be exported: *save sample "sample_587.txt"*
  - Use load sample *"sample_587.txt"* to load the sample back
  - Compounds are not split into elements by default, use command *split sample elements* to do so

Example: sample file

- Example: mixture of oxides



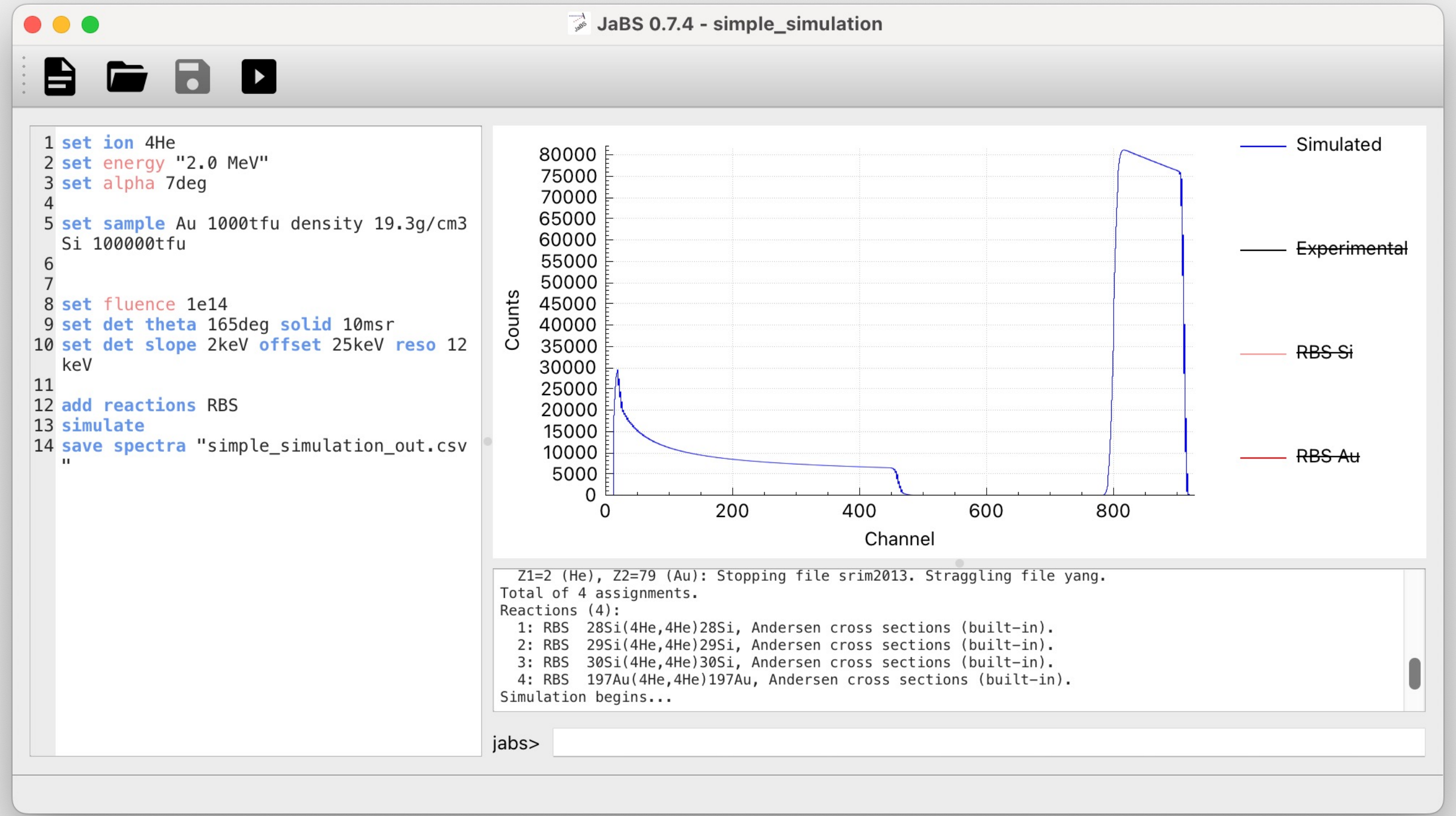
```
thick    NiO Fe2O3    Si   rough
130        1    3     0    10
10000      0    0     1    0
```

- First column should be **thick** when a layered model is used, alternatively **depth** or **width** are used with point-by-point model
  - Difference between depth and width is that width is a difference in depth, see the ”*point_by_point_profile.jbs*” example
- Other keywords (rough, n_rough, yield, yield_slope, bragg, stragg) are optional and they may be in any order as any other column
- If the column is not a keyword, it is assumed it is a material (e.g. *SiO2* or *Si*)

Performing a simulation

- Once everything has been set, just issue command *simulate* (or just *sim* between friends)
- JaBS will do the following:
  - Add reactions, if none have been added
  - Load stopping

Example script: simple simulation



```
set ion 4He
set energy "2.0 MeV"
set alpha 7deg

set sample Au 1000tfu density 19.3g/cm3 Si 100000tfu


set fluence 1e14
set det theta 165deg solid 10msr
set det slope 2keV offset 25keV reso 12 keV

add reactions RBS
simulate
save spectra "simple_simulation_out.csv"
```

```
  Z1=2 (He), Z2=79 (Au): Stopping file srim2013. Straggling file yang.
Total of 4 assignments.
Reactions (4):
  1: RBS  28Si(4He,4He)28Si, Andersen cross sections (built-in).
  2: RBS  29Si(4He,4He)29Si, Andersen cross sections (built-in).
  3: RBS  30Si(4He,4He)30Si, Andersen cross sections (built-in).
  4: RBS  197Au(4He,4He)197Au, Andersen cross sections (built-in).
Simulation begins...
```

- Note use of abbreviations and combining multiple parameters in one command
- Note use of units and quotes
- Order of commands matters a bit
  - sample must be set before adding reactions other things are "set" before simulating
  - spectra can be saved only after simulation
- *add reactions RBS* is optional
- what hasn't been explicitly set is kept as default, e.g. detector is assumed to be in IBM geometry

Load experimental data

- Experimental data can be 1-column, 2-column or arbitrary N column data
  - Use *set detector column 0* to load 1-column data
  - Use *set detector column 1* to load 2-column data (default)
  - This can be confusing because of multidetector support
- Multidetector support: *set detector **N** column **M***
  - first column in file must be the channel, numbering ***M*** starts after this from 1
    - i.e. the channel column is the zeroth (0) column!
    - That's why the default column number is 1 (see *show det)* for 2-column data
    - And of course because with this numbering **N** = **M**
  - all detectors load data from the same file: *load experimental **file***
  - just one detector: *load experimental **N file***
- Columns are separated by whitespace (spaces, tabs) or with commas (,)
  - Comma separated values (CSV) mode is assumed if the filename ends with **.csv**
  - Counts are floating point numbers, exponential notation is supported, decimal separator is dot (.)

Controlling output – verbosity

- JaBS produces “messages” as its output (bottom right hand corner in GUI)
- These messages have 6 levels of verbosity, lower number means higher verbosity:
  - Debug (0)
  - Verbose (1)
    - Some additional (useful?) information is produced
  - Default (2)
  - Important (3)
  - Warning (4)
    - Messages will be yellow in GUI, run will not be aborted
  - Error (5)
    - Messages will be red in GUI, run will be aborted
- Command line (--verbose), GUI Preferences or command “*set verbosity*” can be used to change verbosity

Fitting

- Levenberg – Marquardt algorithm
  - Very commonly used, efficient, finds the “local” solution
- Make initial guess (simulate)
  - For nice samples your first guess might be close enough
- Add fit ranges
  - Note multidetector notation
- Issue fit command
- Evaluate results
- Save results

Fitting

- Several parameters can be fitted
  - Beam energy, sample tilt, layer thicknesses, concentrations…
  - See list with *show fit variables <pattern>*, where optional pattern can be used to show only some variables
    - Wildcards (*, ?) accepted in pattern
    - Examples of variables: fluence,thick2, conc3_Si, det1_calib_solid
  - Note that available fit variables change as sample and detectors are set!
  - *fit var1,var2,var3…*
    - Same pattern matching as before
  - Typical fit command: *fit fluence,*calib*,thick1*
- Fitting calibration and fluence is almost always recommended
  - If you don't give any "wiggle room" for the fitting algorithm, nothing will fit
- Algorithm will start with the initial guess you give, so run *simulate* before fitting
  - How close is close enough? Depends on the problem…

Fitting

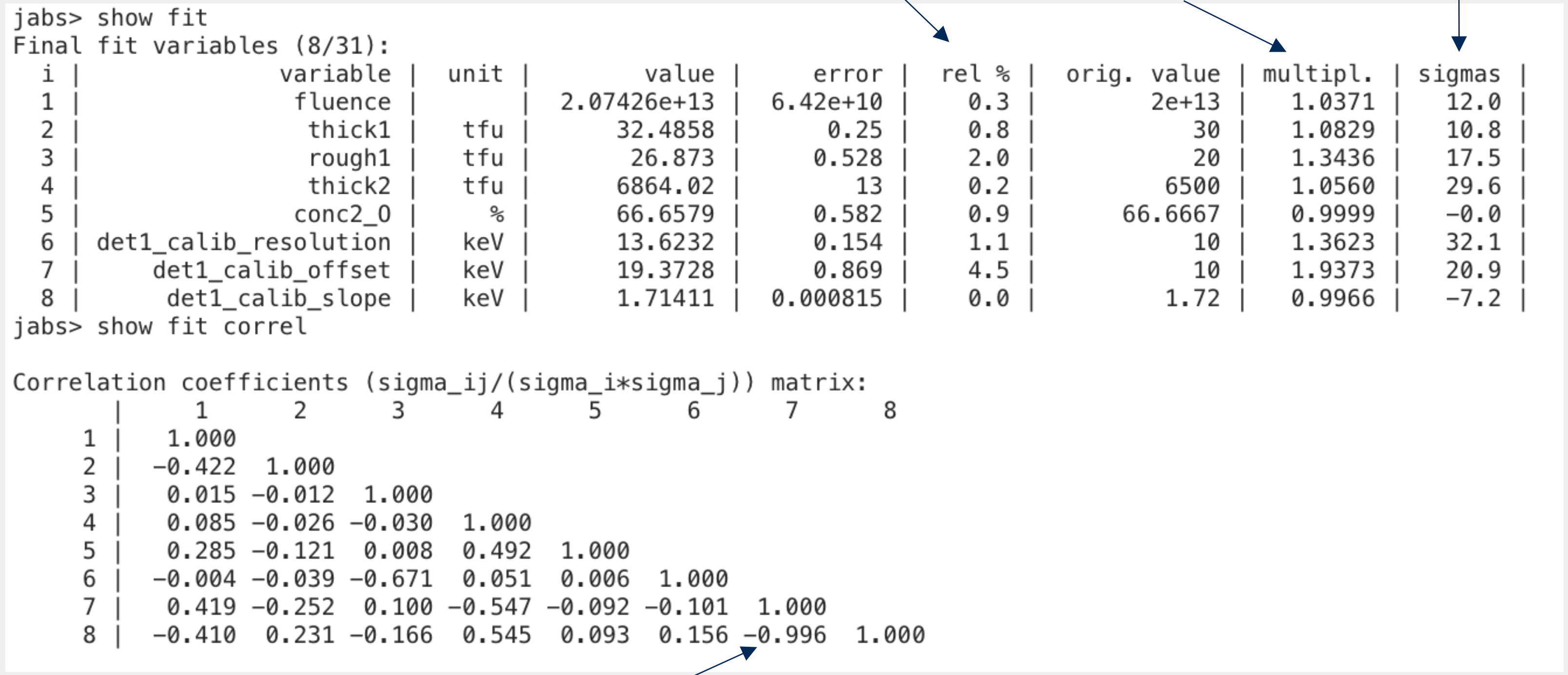
- Two phase fitting (default, *set fit normal*)
  - First phase uses fast simulation model (equivalent to *set sim fast*)
  - Second phase uses the parameters you have set (defaults set by default), starting from the results of the first phase
  - Saves time
- One phase fitting
  - Slow fit (*set fit slow*) skips the first phase
    - Useful if the fast phase deviates too much from initial guess due to the approximations used, for example when there are resonances in cross sections
  - Fast fit (*set fit fast)* stops after the first phase
    - Accurate enough for a lot of work, saves time, not recommended for final values
- On fit failure original values are restored, on success fitted values are kept
  - You don’t have to stop after the first fit command

Evaluation of fit results

How many fit errors (standard deviations) the change was, you can judge the significance

Relative error

Change from original value (multiplier)



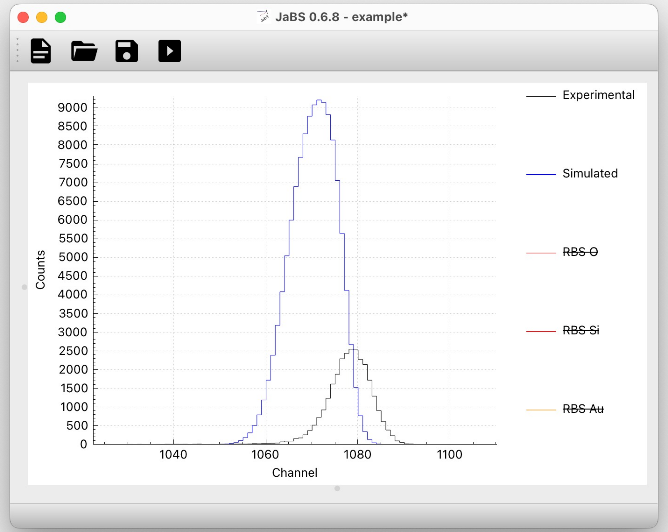
```
jabs> show fit
Final fit variables (8/31):
  i |              variable |  unit |         value |      error |  rel % |  orig. value | multipl. | sigmas |
  1 |               fluence |       | 2.07426e+13 |   6.42e+10 |    0.3 |        2e+13 |   1.0371 |   12.0 |
  2 |                thick1 |   tfu |     32.4858 |       0.25 |    0.8 |           30 |   1.0829 |   10.8 |
  3 |                rough1 |   tfu |      26.873 |      0.528 |    2.0 |           20 |   1.3436 |   17.5 |
  4 |                thick2 |   tfu |     6864.02 |         13 |    0.2 |         6500 |   1.0560 |   29.6 |
  5 |               conc2_O |     % |     66.6579 |      0.582 |    0.9 |      66.6667 |   0.9999 |   -0.0 |
  6 | det1_calib_resolution |   keV |     13.6232 |      0.154 |    1.1 |           10 |   1.3623 |   32.1 |
  7 |     det1_calib_offset |   keV |     19.3728 |      0.869 |    4.5 |           10 |   1.9373 |   20.9 |
  8 |      det1_calib_slope |   keV |     1.71411 |   0.000815 |    0.0 |         1.72 |   0.9966 |   -7.2 |
jabs> show fit correl

Correlation coefficients (sigma_ij/(sigma_i*sigma_j)) matrix:
      |      1      2      3      4      5      6      7      8
    1 |  1.000
    2 | -0.422  1.000
    3 |  0.015 -0.012  1.000
    4 |  0.085 -0.026 -0.030  1.000
    5 |  0.285 -0.121  0.008  0.492  1.000
    6 | -0.004 -0.039 -0.671  0.051  0.006  1.000
    7 |  0.419 -0.252  0.100 -0.547 -0.092 -0.101  1.000
    8 | -0.410  0.231 -0.166  0.545  0.093  0.156 -0.996  1.000
```

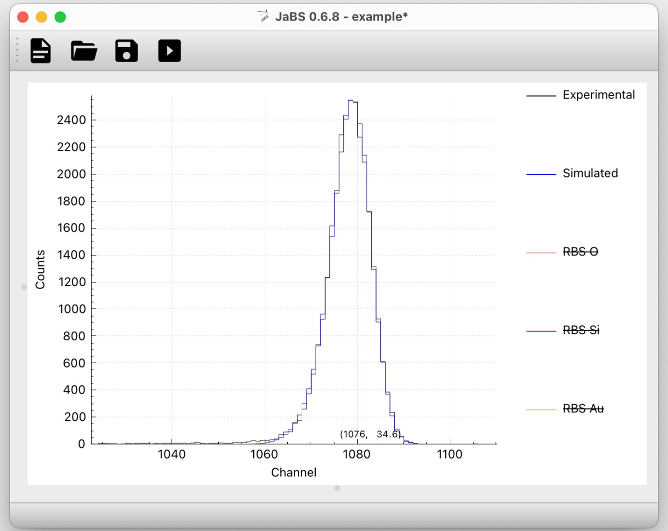
- Values are converted to some unit
- Fit parameter error (standard error) estimated by the L-M algorithm is a measure of how much does changing the parameter affect the fit. Statistical measurement uncertainty is used as weights in the L-M algorithm. Useful as-is for uncertainty analysis, but note that “physics” of JaBS does not have any uncertainty and therefore e.g. wrong stopping forces don’t contribute to uncertainty!
- Errors can be correlated, remember this if you are doing uncertainty analysis of derived quantities (like ratios of concentrations).

Note strong negative correlation between calibration slope and offset as one would expect!

Example script: simple fit

- Provided example.jbs
  - Fitting thin Au / SiO_2
- Initial guess rule of thumb:
  - **"Peaks have to overlap"**
  - Include enough "nothing" around peaks in fit ranges
- L-M goes where the derivatives point!
  - Fluence is usually not an issue
  - Calibration needs to be close, especially in case of narrow peaks. Calibration offset of zero is a problem, since all changes are relative to initial value.
  - Leave roughness and concentrations out if the fit fails, improve your initial guess (based on this fit) and retry.
- Fitting rule of thumb:
  - "**Start with a simple model and refine it**"
- Not an AI
  - Layers can not change order, thickness can go down to zero
  - If fit is aborted by sanity checker, L-M can't help you. Improve your guess, try again.





IBA data format (IDF) support

- Converter from IDF is included
  - GUI: File | Import
  - Command line: *jabs idf2jbs* ***file***
- Works by creating appropriate *set* commands from the XML file
- Support is partial, but reasonably useful
  - Layered samples
  - Energy and ToF detectors
  - Geometrical broadening (beam spot, detector size)
  - Experimental and simulated spectra
  - Multiple spectra / samples per file
  - Missing: physics defaults, handling of reactions
- Import of IDF files (*.idf, *.xml) exported from SIMNRA as well as *.xnra files as-is
- Export of a simulation (and experimental data) using a command: *save sim* ***file***

Notes on GUI

- Isotope spectra are automatically summed
  - See preferences to enable separate isotope spectra of either all elements or one element
  - Summing of similar reactions only
    - For example built-in RBS and R33 file spectra will be shown separately
- Keyboard shortcuts exist
  - Run is F5 (Windows), Ctrl + R (Linux), ⌘+R on macOS
- Many ways to manipulate the plot window
  - Plot shortcuts, lowercase letters work too:
    - ***W***, ***S***, ***A***, ***D*** move up, down, left, right respectively. Arrow keys do the same.
    - ***1*** toggles vertical autorange (visible range fills vertical range, on by default)
    - ***X*** or ***0*** reset zoom
    - ***Z*** toggles zoom mode (rectangle)
    - ***Q*** and ***E*** change vertical range by a factor of two when autorange is disabled
    - ***L*** toggles logarithmic vertical scale
    - **R** toggles range selection, runs ROI command and copies range to clipboard when finished
  - See context menu by clicking with right mouse button on top of the plot.
  - Both axises can be dragged, mouse wheel zooms in/out both axes at the same time or one axis at a time when hovering on top of an axis

Intermediate difficulty use of JaBS 🙂

Beyond basic RBS: ERD, EBS, NRA. Detectors, roughness and other complications.

Reactions and defining cross sections

- R33 file support
- Built-in cross sections, Rutherford as-is or with Andersen screening
  - If no reactions are defined the default set of reactions is added:
    - RBS always
    - RBS (–) solution if possible
    - ERD in forward angles
  - Use *add reactions RBS* to explicitly add RBS for all target isotopes
    - Sample must be defined first! Setting a new sample resets reactions.
    - Substitute “RBS-” or “ERD” for those reactions
  - These can be limited to an energy interval to complement reactions from R33 files
- Arbitrary cross sections possible with a plugin (see advanced use)

Example: non-Rutherford scattering $^{16}O(^4He,^4He)$

1. Acquire cross section data somewhere (experimental, SigmaCalc)
2. Use *“load reaction somefile.r33”*

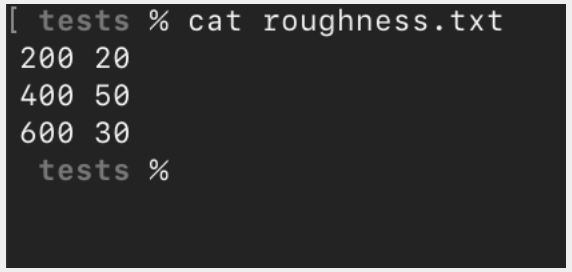
- JaBS will tell you what reaction you just loaded and what is the range of energies of the cross sections in this file. For example: `Added reaction 8 (FILE 16O(4He,4He)16O), E = [1769 keV, 6890 keV], Q = 0 keV`

1. Add RBS reactions manually: *add reactions RBS*
2. The ^{16}O reaction loaded by the previous step needs to be removed, since there is an overlap in energy above 1769 keV, so we issue command: *remove reaction RBS 16O*
3. If ^{16}O RBS spectra are required when the ion energy is below 1769 keV we should add a new reaction: *add reaction RBS 16O cs Andersen max "1.769 MeV”*
4. If “Theta” angle in R33 file does not match detector theta, reaction is not calculated. Default tolerance 1 degree.

See file “non_rutherford.jbs” in JaBS examples.

Roughness (thickness variation)

- Substrate roughness is not implemented
- Simulation of subspectra with varying thickness, note correlation between incident and exiting ion
- Default roughness model is gamma roughness
  - Amount of roughness is a parameter that can be fitted
- Arbitrary roughness can be loaded from a file using *load roughness <layer number> <file>*
  - Note that when saving sample (*save sample*) this data is not included
  - Files look a bit different to those used by SIMNRA
    - Example below would simulate three spectra, 20%, 50% and 30% weight on each (JaBS will normalize…), with 200 tfu, 400 tfu and 600tfu thicknesses, respectively
    - Continous thickness distribution (e.g. gaussian…) must be turned discrete by yourself



```
[ tests % cat roughness.txt
200 20
400 50
600 30
 tests %
```

Ad-hoc correction factors

- Several layer (depth range) specific correction factors are available
  - Stopping force multiplier “*bragg*”
    - Example: *set layer SiO2 1000tfu bragg 1.1*
  - Straggling multiplier “*stragg*”
  - Yield multiplier “*yield*”
    - Yield slope “*yield_slope*”, total yield is calculated at depth *x* from the beginning of layer (closer to surface) using equation: $Y(x) = yield + yield\_slope \times x$, where *x* is always assumed to be in units of tfu (10^{15} at./cm^2), i.e. no unit should be given for *yield_slope*.
    - *set channeling yield* and *slope* commands set the last layer yield and yield_slope, respectively
      - The corresponding fit parameters are *channeling* and *channeling_slope*
  - Defaults are, of course, 1.0 for bragg, stragg and yield, 0.0 for yield_slope
  - Correction factors are automatically omitted from output if they are equivalent to the defaults

Detector foil

- *set det foil <foil>*, where foil is a standard (multilayer) sample description
  - Order of layers is the order they are encountered by reaction products, the first layer is “on top”, the last is closest to detector
  - Roughness is not calculated, straggling is
- Example: *set det foil Si 150tfu* sets 30 nm silicon (dead layer)

Detector calibration for different reaction products (elements)

- Different elements can have completely independent calibration, including resolution
- Example: *set det calibration H linear slope 1.5keV offset 2keV resolution 10keV*
- Important to specify “linear” or “poly”. That triggers the creation of a calibration override for this specific element

JYVÄSKYLÄN YLIOPISTO
UNIVERSITY OF JYVÄSKYLÄ

Advanced use of JaBS 🤓

Things you don’t need to know, but somebody will ask anyways

Multidetector use

- Intended use is multidetector RBS
  - Calibration of multiple detectors with the same scattering angle
  - Simulation and fitting of spectra from multiple detectors in different geometries
- May also be useful with conventional ERD with simultaneous RBS
- Use *add detector default* to add a new detector with default configuration
- Use *det detector N* instead of *set detector* to set properties of a particular detector
  - N is a number between 1 and number of detector
    - Words *first* and *last are* interpreted as the first and last detector, respectively
    - Detector name may also be used (given *by set detector name*)
  - When JaBS starts, one default detector is defined
    - Use *reset detectors* to remove all detectors, including the default
- Top tip: make a script to generate detector files

Assigning custom stopping forces

- For each Z_1, Z_2 combination stopping data is required
- JIBAL is in charge of stopping data files
  - One file may include ranges of both Z_1 and Z_2
  - Binary distributions for macOS and Windows include some stopping and straggling files
  - JIBAL has a file (**files.txt**) that determines which file will be used for which combination
    - Default is SRIM data for all electronic stopping ($1 \leq Z \leq 92$) and Yang for straggling $2 \leq Z_2 \leq 97$
- For the duration of a session JaBS does not reinitialize JIBAL
  - Files can not be added while JaBS is running and the contents should not be changed
- Use *set stopping* to override JIBAL default assignment
  - Example: *set stopping dpass He Si* if you have electronic stopping data from DPASS in **dpass.ele**
  - Used assignments and filenames are printed when simulating
- It is possible to set up JIBAL configuration to provide data outside the distribution
  - See jibal.conf, JIBAL bootstrap or *jibaltool (.exe)*

Calculating confidence limits

- 95% confidence limits of fitted spectra can be calculated (reflect the errors of fit parameters in spectra) can be calculated by *set cl true*, which should be issued before fit.
- The calculations are probably wrong, so as of JaBS 0.7.5 they can not be trusted. Qualitatively they seem to be reasonable.
- The +/- confidence limits are plotted in the GUI, but can be disabled
- Saving of CL spectra using *save spectra* follows also the *cl* parameter. You can also *set cl false* before saving.

Summary

And final remarks

Give it a go

- JaBS **is** ready for serious use
  - No warranty (don't sue me if it doesn't work right)
- It is **not** difficult to use
  - Import an IDF file, look at examples
- Lightweight
  - Qt is slightly bloated
  - Command line executable in macOS just 300 kB! Typical memory use (peak) during RBS fit measured around 5 MB. Will work on almost any computer made this century.
- GPL license means you can use it for any use, including commercial use, modify and redistribute it
  - Conditions apply; please read the license for details
- Free to download, free to use, no license fees, can be distributed and modified under certain conditions
- No advertising, no telemetry, no data is collected of the user, no spying, no GDPR issues

Known issues

- Version 0.7.4 (Oct 14, 2024)
- Version 0.7.3 (Oct 5, 2023)
  - Hitting the maximum number of bricks did not produce any output at all
  - Using Z specific detector calibrations in fits caused crashes
  - IDF export: polynomial calibration wasn't saved correctly
- Version 0.7.2 (Sep 29, 2023)
- Version 0.7.1 (Apr 10, 2023)
  - Crash when simulating element equal to highest Z_2 in stopping data (default settings: 92, i.e. U). Bug since 0.7.0.
- Version 0.7.0 (Feb 27, 2023)
  - Parsing of "yield_slope" correction from sample files overwrites "yield" as well, use of sample files with yield_slope correction is therefore impractical
  - Calculation of maximum depth (of an isotope) had a bug, at least when concentration gradients that fall down to pure zero were used.
  - JaBS will crash when samples with more than 8 layers are set
- Version 0.6.9 (Unreleased)
- Version 0.6.8 (Feb 6, 2023)
  - Beam energy fitting didn't work. Binary versions of "0.6.8" are actually one commit ahead of tagged "v0.6.8" version and this is fixed in them.
  - Yield slope and "channeling" ad-hoc corrections not implemented
  - Command *roi* can divide by zero and doesn't fail on if rois are set backwards, e.g. [200:100]. GUI tool can make these kinds of regions of interest.
- Earlier versions
  - See list of changes at https://github.com/JYU-IBA/jabs/releases