Superflip — A computer program for solution of crystal structures from x-ray diffraction data in arbitrary dimension.

User Manual

written by Lukáš Palatinus
palat@fzu.cz

Laboratoire de Cristallographie
Le Cubotron
Ecole Polytechnique Féderale de Lausanne
1015 Lausanne

version: 31/08/2007

Contents

1 Introduction **4**

2 Theoretical background **5**
2.1 Generalized iterative algorithm 6

3 Installation and execution **8**
3.1 Compilation of SUPERFLIP 8
3.2 FFTW library 8
3.3 Execution 9

4 Format of the ASCII input file **10**
4.1 Specification of keywords 10
4.1.1 name: addcycles 11
4.1.2 name: bestdensities 11
4.1.3 name: biso 11
4.1.4 name: cell 11
4.1.5 name: centers 12
4.1.6 name: commandfile 12
4.1.7 name: composition 13
4.1.8 name: convergencemode 13
4.1.9 name: coverage 13
4.1.10 name: dataformat 14
4.1.11 name: dataitemwidths 15
4.1.12 name: delta 15
4.1.13 name: derivesymmetry 16
4.1.14 name: dimension 16
4.1.15 name: expandedlog 17
4.1.16 name: fastfft 17
4.1.17 name: fbegin (– endf) 17
4.1.18 name: filebase 18
4.1.19 name: finevoxel 18
4.1.20 name: fullreflections 18
4.1.21 name: fwhmseparation 18
4.1.22 name: histogram (- endhistogram) 19
4.1.23 name: hmparameters 20
4.1.24 name: lambda 20
4.1.25 name: maxcycles 20
4.1.26 name: missing 20
4.1.27 name: modelfile 21
4.1.28 name: modelformat 21
4.1.29 name: normalize 21
4.1.30 name: nresshells 22

4.1.31 name: outputfile . . . 22
4.1.32 name: outputformat . . . 22
4.1.33 name: perform . . . 22
4.1.34 name: polish . . . 23
4.1.35 name: presentationmode . . . 24
4.1.36 name: qvectors – endqvectors . . . 24
4.1.37 name: randomseed . . . 24
4.1.38 name: realdimension . . . 24
4.1.39 name: referencefile . . . 24
4.1.40 name: referenceformat . . . 25
4.1.41 name: reflectionlist . . . 25
4.1.42 name: reflendline . . . 25
4.1.43 name: reflstartline . . . 26
4.1.44 name: repeatmode . . . 26
4.1.45 name: reslimit . . . 26
4.1.46 name: resunits . . . 26
4.1.47 name: rewriteoutput . . . 27
4.1.48 name: searchsymmetry . . . 27
4.1.49 name: skipstartcycles . . . 27
4.1.50 name: symmetry – endsymmetry . . . 27
4.1.51 name: terminal . . . 28
4.1.52 name: testsymmetry – endtestsymmetry . . . 29
4.1.53 name: title . . . 29
4.1.54 name: usephases . . . 29
4.1.55 name: viewprogress . . . 29
4.1.56 name: voxel . . . 30
4.1.57 name: weakratio . . . 31
4.2 Examples of an input file . . . 31
4.2.1 A minimalistic example . . . 31
4.2.2 A realistic example for a modulated structure . . . 32

5 Description of the output **33**
5.1 The electron density file . . . 33
5.1.1 format xplor . . . 33
5.1.2 Format ccp4 . . . 34
5.1.3 Format jana . . . 34
5.1.4 Format m80 . . . 34
5.2 The log-file . . . 34
5.2.1 The concise form of the log file . . . 34
5.2.2 Information about the input . . . 34
5.2.3 Information about the processing of the reflections . . . 35
5.2.4 Information about the iteration . . . 36

6 Run-time interaction with the program **38**

7 Handling of symmetry in Superflip **39**
7.1 Automatic derivation of symmetry from the density . . . 40

8 Superflip and powder data **43**

9 Some know-how or what to do if things go wrong **45**
9.1 The value of δ 45
9.2 The symmetry 46
9.3 The convergence 46
9.4 Charge flipping converges, but I cannot refine the structure! 46

Chapter 1

Introduction

The solution of the phase problem in crystallography is the central issue in the process of recovering structure information from x-ray diffraction data. Thanks to the development of direct methods, this problem can be solved routinely now for a large part of small- to medium-sized structures. Nevertheless alternative methods still can make a decisive contribution to solution of structures that are difficult or impossible to solve by direct methods and related techniques.

Charge flipping is an iterative algorithm for reconstructions of approximate electron densities from structure factor amplitudes. While it probably does not supersede direct methods in terms of the maximum size of solvable structures, it is superior in terms of the amount of prior information required. While most other structure solution methods are based on the assumption of the atomicity of the structure and the symmetry is also usually exploited, charge flipping needs neither symmetry nor the atomic character. This makes it particularly suitable for structure solution of modulated structures and quasicrystals, where the atoms form continuous domains in a (3+d)-dimensional space.

Superflip (the name stands for charge *flip*ping in *super*space) is a computer program that has been written to provide an effective tool for applications of the charge flipping algorithm. It allows for an (almost) automated structure solution of simple structures as well as a detailed control and advanced options for investigation of difficult structures.

Chapter 2

Theoretical background

The algorithm dubbed charge flipping has been described in two papers by Oszlány and Sütő (Oszlányi and Sütő, 2004; Oszlányi and Sütő, 2005). The reader interested in the details of the algorithm and the deeper theoretical background is strongly advised to study these two articles. The extension of charge flipping towards solution of incommensurately modulated structures has been described in Palatinus (2004).

In this manual only the basic concept of the algorithm will be outlined together with the basic terms that occur further in the manual.

The electron density ρ is sampled on a grid with $N_{pix} = N_1 \times N_2 \times N_3$ pixels. The density values ρ_i are evaluated in each pixel $i = 1, \ldots, N_{pix}$ of the grid. $|F^{obs}(\mathbf{H})|$ are the experimental amplitudes of the structure factors. The algorithm is initiated in the zeroth cycle by assigning random starting phases $\varphi_{rand}(\mathbf{H})$ to all experimental amplitudes and making all unobserved amplitudes equal to zero:

$$F^{(0)}(\mathbf{H}) = \begin{cases} |F^{obs}(\mathbf{H})| \exp\left(i\varphi_{rand}(\mathbf{H})\right) & \text{if } |F^{obs}(\mathbf{H})| \text{ is known} \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

The iteration cycle then proceeds as follows:

1. The density $\rho^{(n)}$ is calculated by inverse Fourier transform of $F^{(n)}$.
2. The modified density $g^{(n)}$ is obtained by flipping the density of all pixels with density values below a certain positive threshold δ and keeping the rest of the pixels unchanged:

$$g_i^{(n)} = \begin{cases} \rho_i^{(n)} & \text{if } \rho_i^{(n)} > \delta \\ -\rho_i^{(n)} & \text{if } \rho_i^{(n)} \leq \delta \end{cases} \tag{2.2}$$

3. Temporary structure factors $G^{(n)}(\mathbf{H}) = |G^{(n)}(\mathbf{H})| \exp\left(i\varphi_G(\mathbf{H})\right)$ are calculated by Fourier transform of $g^{(n)}$.
4. New structure factors $F^{(n+1)}$ are obtained by combining the experimental amplitudes with the phases φ_G and setting all non-measured structure factors to zero:

$$F^{(n+1)}(\mathbf{H}) = \begin{cases} |F^{obs}(\mathbf{H})| \exp\left(i\varphi_G(\mathbf{H})\right) & \text{if } |F^{obs}(\mathbf{H})| \text{ is known and strong} \\ |G^{(n)}(\mathbf{H})| \exp\left(i(\varphi_G(\mathbf{H}) + \pi/2)\right) & \text{if } |F^{obs}(\mathbf{H})| \text{ is known and weak} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

In the standard variant of the algorithm no reflections were treated as weak. In the improved variant (called the "π-half" variant sometimes) the reflections are sorted by their amplitudes and certain proportion of the smallest amplitudes is considered to be weak.

These modified structure factors then enter the next cycle of iteration.

The $F(\mathbf{0})$ structure factor is set to zero in the zeroth cycle of the iteration and allowed to change freely in the subsequent cycles.

δ is the only adjustable parameter of the algorithm. Its value should be selected small relatively to the maximum density, but larger than the typical amplitude of the Fourier artifacts induced by the series termination error. In practice, the value of δ is determined by trial and error.

An important aspect of the algorithm is that all operations are performed in the whole unit cell with symmetry $P1$. The origin of the structure is thus not fixed and the structure can emerge anywhere in the unit cell.

The progress of the iteration can be monitored for example by observing the R-value of amplitudes $|G^{(n)}(\mathbf{H})|$ with respect to $|F^{obs}(\mathbf{H})|$. It is large in the initial cycles of the iteration, and the onset of the convergence is signalled by a sharp decrease of the R-value. The iteration is converged, if the R-value stops decreasing and oscillates around a constant value. The final R-values are larger than the values typical for successful structure refinement, typically 20-30%. However, the R-value is not used as a measure of the quality of the reconstruction, but merely as an indicator of convergence.

The generalization of the algorithm for reconstructions of incommensurately modulated and composite crystal structures is straightforward. Following the method of embedding of aperiodic crystal structures in superspace, the $3D$ density is replaced by a $(3+d)D$ superspace density sampled using a $(3+d)D$ grid with $N_{pix} = N_1 \times N_2 \times \ldots \times N_{3+d}$ pixels, where d is the number of independent modulation vectors. The structure factors are indexed by $(3+d)$ integer indices. They represent the coefficients of the Fourier transform of the superspace density. With these modifications, the algorithm described at the beginning of this section can be applied directly to incommensurate structures.

2.1 Generalized iterative algorithm

In reality the charge flipping algorithm can be considered as one of the family of iterative algorithms, which includes also the Fienup's hybrid input-output algorithm, Elser's Difference Map, and other variants. All these algorithm (in it's basic form, without the extensions like the handling of the weak reflections) can be understood as special cases of a general algorithm that can be written in the form:

$$\rho_{(n+1)} = [(1-\beta_1-\beta_2)\mathbf{I} + \beta_1 \mathbf{T}_R^{\gamma_{R1}} \otimes \mathbf{T}_D^{\gamma_{D1}} + \beta_2 \mathbf{T}_D^{\gamma_{D2}} \otimes \mathbf{T}_R^{\gamma_{R1}}]\, \rho_n \tag{2.4}$$

Here subscripts R and D refer to "reciprocal space" and "direct space", β and γ are adjustable parameters, $\mathbf{I}$ is the identity, and $\mathbf{T}^\gamma$ is an overprojection defined as

$$\mathbf{T}^\gamma = (1+\gamma)\mathbf{P} - \gamma\mathbf{I} \tag{2.5}$$

where $\mathbf{P}$ s the corresponding projection. In reciprocal space the projection $\mathbf{P}_R$ is defined by equation 2.3 or some of its variants, in direct space the projection $\mathbf{P}_D$ is a density modification equal to or similar to the simple positivity projection:

$$g_i^{(n)} = \begin{cases} \rho_i^{(n)} & \text{if } \rho_i^{(n)} > \delta \\ 0 & \text{if } \rho_i^{(n)} \leq \delta \end{cases} \tag{2.6}$$

Note that equation 2.2 amounts to an overprojection $\mathbf{T}_D^1 = 2\mathbf{P}_D - \mathbf{I}$. Note also that the projections need not necessarily be true projections in the mathematical sense, but can be replaced by more general mappings, provided the basic properties of convergence and stability are preserved. Such a generalized mapping is for example the π-half variant of the reciprocal-space modification.

Superflip contains an implementation of this general algorithm, and internally the charge flipping algorithm is just a special case of the general algorithm with $\beta_1 = 1$, $\gamma_{1R} =$

0, $\gamma_{1D} = 1$, and $\beta_2 = 0$. Other algorithms can be obtained by setting the six variable parameters differently. Note however that not all combinations of the six parameters give a valid algorithm that converges!

Chapter 3

Installation and execution

3.1 Compilation of SUPERFLIP

The program is written in standard Fortran 95 and as such it should be compilable with any f95 compiler. The only exception is an occasional use of a subroutine `system`, but this extension to standard F95 is available in all compilers known to me. Most providers of unix workstations provide their own compiler suite. In addition to that, free fortran 95 compilers are now available (`http://gcc.gnu.org/wiki/GFortran`),`www.g95.org`).

The distribution package should be unpacked in a separate directory. The header of the `Makefile` must be modified manually if the commands calling the fortran and c-compilers are different from the default (currently gfortran and cc, respectively), or if the compiler options are not suitable for the given platform. The program is compiled by running `make` from the command line:

```
$ make
```

Prior to running make, the FFTW3 library has to be installed on the system and available to the linker (see below). After the compilation passes successfully, the resulting executable named `superflip` should be copied to a desired location, preferable one that is contained in the system variable PATH.

3.2 FFTW library

In order to speed up the performance of the program, `superflip` uses the library FFTW for computing of the discrete Fourier transform. Corresponding library (version FFTW 3.X or higher) must be therefore installed on your system prior to compiling the program itself.

The source codes for FFTW are available at www.fftw.org. Download the source codes and unpack to a separate directory. The compilation consists of three steps. First run the configure script:

```
$ ./configure --enable-float
```

The option `--enable-float` is necessary, because the default precision of the FFTW is double precision, but SUPERFLIP works in single precision (which, however, does not make the results less precise...).

Attention! After running `configure`, always check the file `config.log` for a string `disable-fortran` by typing:

```
$ grep "disable-fortran" config.log
```

On some systems the configure script has difficulties in figuring out the way to link fortran and C programs. This results in disabling the fortran interface to the C-routines. A message about disabling this interface is written in the file `config.log`. Thus, if `config.log` contains the string `disable-fortran`, measures have to be undertaken to allow configure to find the way to link fortran and C. Currently, this problem is known to occur on Mac OS X. The file `config.log` contains a bunch of error messages starting with `"/usr/bin/ld: multiple definitions of symbol "`. The solution is to replace following line in the file `configure`:

```
-lang* | -lcrt0.o | -lc | -lgcc | -libmil | -LANG:=*)
```

by line:

```
-lang* | -lcrt[012].o | -lc | -lgcc | -libmil | -LANG:=*)
```

After a successfull run of configure the library is compiled by running make:

```
$ make
```

and the compiled libraries are exported to their destination (usually /usr/lib) by typing

```
$ make install
```

You will need the administrator's access rights to do the last step.

If the libraries are placed in a standard location, the linker will usually find them without problems. However, if the location is non-standard (for example: you do not have the administrator's rights and want to install FFTW only on your account), you have to tell the linker where to find the library. This can be done by modifying the following line in the Makefile:

```
linklib = -lfftw3f -lm
```

to

```
linklib = -Lpath_to_FFTW -lfftw3f -lm
```

where "`path_to_FFTW`" is a full path to the location of the library libfftw3f.a.

3.3 Execution

The program is executed from the command line by typing:

```
$ superflip [--version] filename [maxcycles]
```

Filename is the name of the ASCII input file containig the instructions for the program. The optional argument maxcycles defines the maximum cycles of the iteration. If the calculation does not converge within maxcycles, the iteration is interrupted. If maxcycles is omitted, it is set to a default value 10000.

If SUPERFLIP is invoked with option `--version`, it prints out the current version number – or more precisely the date and time of the creation of that version, and quits.

Chapter 4

Format of the ASCII input file

The input file is a free-format ASCII file based on keywords. Each keyword represents a specific command or parameter for the program and must be given a value.

Multiple spaces anywhere in the file are handled as a single space. If the character `'#'` or `'!'` occurs anywhere in the line, the rest of the line after this sign is treated as comment and not interpreted. Blank lines anywhere in the input file are ignored. The length of the interpreted part of the line is 132 characters, any text exceeding this length is ignored.

4.1 Specification of keywords

There are two basic types of keywords. The first type is followed by one or more values on the same line:

```
keyword value1 [value2 value3...]
```

The second type has the form:

```
initial_keyword
line 1
line 2
...
final_keyword
```

Each line may contain one or more values.

The name of the keyword of the first type is a single word without spaces. The name of the keyword of the second type is a pair of an initial and a final word (separated by a hyphen in the following text).

Each value can be a constant of type real, integer or character. The type of the parameters and their allowed values are specified. Alternative values are separated by slashes.

The keywords are either compulsory or optional. The compulsory keywords must be specified for the analysis to proceed. The optional keywords can be omitted. If an optional keyword is omitted, the default value is used. Compulsory keywords are indicated by "compulsory keyword – no default value" in the item "default".

The item "description" describes the function of the keyword, its influence on the output and relations to other keywords.

4.1.1 name: addcycles

- value: positive integer
- default: 0
- desctription: Sometimes the automatic procedure for detection of convergence detects the convergence before the iteration is completely stabilized. This is often the case for macromolecules. If this is the case, it is possible to use this keyword to force SUPERFLIP to add a number of additional cycles (typically 100 - 300) to the iteration after the convergence has been detected.

4.1.2 name: bestdensities

- value: positive integer [rvalue/peakiness/symmetry/reference]
- default: 1 rvalue
- desctription: SUPEFLIP has the option to repeat the calculation. See keyword `repeatmode` to learn more about this option. If the calculation is repeated, the program saves the densities with the best figure of merit. If bestdensities = 1, only one density with the name given by the keyword `outputfile` is saved. This density is rewritten each time a new density has a better figure of merit than the saved density. If bestdensities > 1, the corresponding number of best densities are saved with the name `bestXX_`*outputfile*, where `XX` stands for a serial number of the density and runs from 1 to bestdensities. Outputfile is the filename given as a value of the keyword `outputfile`. The program writes out the list of the properties of the saved densities after each calculation to the logfile and, with the setting `terminal yes`, also to the standard output.

  The figures of merit are currently four: The iteration R-value, the peakiness value, the overall symmetry agreement factor, and (for testing purposes) the agreement factor of the match with the reference file. Particular figure of merit is selected with the second argument of the keyword `bestdensities`.

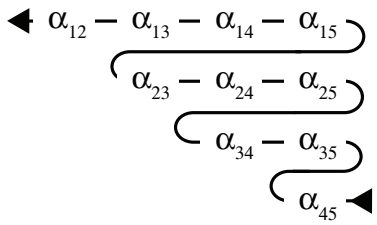
4.1.3 name: biso

- value: positive real number [fix]
- default: 0.0
- description: Defines the overall isotropic Debye-Waller factor B_{iso}. This factor is used to approximately cancel the decrease of the intensities by the thermal motion and effectively sharpens the resulting map. This sharpening can be vital for the success of the algorithm, if $B_{iso} > 4\,\text{Å}^2$ or so. For most inorganic materials B_{iso} is small and need not be taken into account. If the data are normalized using the Wilson plot (see keyword `normalize`), then the B_{iso} derived from the Wilson plot is used for normalization by default. With setting `biso` *number* `fix` this number will be used for normalization instead of the B_{iso} from Wilson plot.

4.1.4 name: cell

- value: a b c α β γ
- default: compulsory keyword – no default value
- description: Gives the dimensions of the unit cell. For two dimensional real space the order of parameters is a b γ, for one-dimensional case only one entry a is expected.

For dimensions of the real space higher than 3 (see keyword `realdimension` for explanation) the lengths of the unit cell dimensions must be listed first, and then the angles between the individual pairs of basic vectors according to following convention (demonstrated on a five-dimensional example):



Here α_{ij} is an angle between the basic vector $\mathbf{a_i}$ and $\mathbf{a_j}$. This convention is a generalization of the convention used for the common 3D case.

4.1.5 name: centers

- value: each line contains one centering vector
- default: zero vector
- description: If the cell is non-primitive, a complete list of centering vectors must be given here. The components of the centering vectors can be either decimal numbers, or fractions with nominator and denominator separated by slash. Examples:

  A-centering:

```
centers
0 0 0
0 0.5 0.5
end centers
```

  R-centering:

```
centers
2/3 1/3 1/3
1/3 2/3 2/3
end centers
```

  As can be seen from the examples, the zero (trivial) centering vector may, but need not be included in the list.

4.1.6 name: commandfile

- value: a valid filename
- default: `jobname.sfcom`
- description: Defines the name of the command file (see chapter 6).

4.1.7 name: composition

- value: Sequence of groups elemental symbol[number of atoms in a unit cell] separated by spaces. Example: Na4 V4 O10
- default: unknown composition
- description: The composition is used, if normalization of the input intensities is desired (see keyword `normalize`. Moreover, if composition is present, the Wilson plot is calculated and printed in the sflog file.

  Element symbol not followed by a number is equivalent to the element symbol followed by 1. The number of atoms should always correspond to the full unit cell and not to the asymmetric unit.

4.1.8 name: convergencemode

- value: normal/rvalue/charge/peakiness [threshold]
- default: normal 0.8
- description: Maybe somewhat surprisingly the detection of convergence of the calculation is an extremely complicated task. Not all calculations obey the classical "plateau – steep change – pleateau" pattern. The default convergence detection in Superflip (`convergencemode normal`) therefore relies on an elaborate combination of analysis of four indicators together with their first and second derivatives. However, in some extreme cases even this analysis does not allow for reliable detection of convergence. This is especially true for the very simple structures, where the convergence is indistinguishable from the initial transition period, and for macromolecules, where the onset of convergence is often too slow to be visible as a peak in the derivatives.

  For these reasons it is possible to base the detection of convergence also on surpassing a threshold value of one of three indicators:

  – `convergencemode rvalue [threshold]`: Convergence is detected, if the iteration R-value drops under the threshold. Default threshold is 30%.

  – `convergencemode charge threshold`: Convergence is detected, if the total charge of the density drops under the threshold. There is no default threshold for this option.

  – `convergencemode peakiness [threshold]`: Convergence is detected, if the value of the peakiness exceeds the threshold. Default threshold is 3.0. Peakiness (in statistics called the skewness of the distribution) is the third central moment of the distribution of the electron density:

$$\gamma = \frac{1}{A}\sum_{i=1}^{N_{pix}}(\rho_i - \bar{\rho})^3 \tag{4.1}$$

  where A is a normalization constant. In Superflip A is set to 1 in first 10 cycles, and to the value of γ at 10^{th} cycle from 11^{th} cycle on.

  The suitable values of the thresholds must be usually first found during a test run, and then set for subsequent calculations.

4.1.9 name: coverage

- value: yes/no
- default: yes

- description: If `yes`, the coverage of the data as a function of resolution is performed. It is very recommended to check this table since a poor coverage might indicate some data inconsistency or simply a too low coverage for charge flipping to work. The calculation is very fast, only for very large problems the calculation of coverage can take considerable time and in that case it can be desirable to switch it off.

4.1.10 name: dataformat

- value: intensity/amplitude/amplitude_difference/a/b/phase/group/dummy/fwhm and combinations of these items OR one of m91/m90/shelx
- default: a b
- description: This keyword defines the format of the reflection entries in the keyword `fbegin - endf`. The keyword `dataformat` must be followed by one or more items from this list:
  - intensity: gives the intensity ($|F|^2$) of the reflection
  - amplitude: gives the amplitude ($|F|$) of the reflection
  - amplitude_difference: gives the difference in amplitudes; applicable for anomalous difference data for determination of heavy-atom substructure. SUPERFLIP takes an absolute value of the input and handles it as an ordinary $|F|$.
  - a, b: define the real and imaginary components of the structure factor
  - phase: Phase of the structure factor. Phase is expressed as a multiple of 2π, i.e. 0.5 corresponds to π, 0.25 corresponds to $\pi/2$. Questionmark (?) can be used in place of an unknown phase.
  - group: Reflection group. Applicable for powder diffraction data, where some reflections are overlapped.
  - fwhm: Full width at half maximum of a reflection in the powder diagram (in units of 2θ). Applicable for powder diffraction data, serves as a basis for calculation of the overlap groups.
  - dummy: Stands for any entry that should be ignored by the program. Can be used for example to make the program skip a column with standard deviations or some other irrelevant entry in the reflection list.

  Each entry in the reflection list (see keyword `fbegin - endf`) is then asssumed to consist of the reflection indices and items corresponding to the list of items after `dataformat`.

  The information in the list must not be redundant, i.e. it is not possible to give amplitude and intensity at the same time, and it is not possible to combine `a`, `b` with either of `intensity`, `amplitude` or `phase`. It is also not possible to combine `fwhm` and `group`. On the other hand, the items `group` or `fwhm` and any number of items `dummy` can be combined with any of the other entries.

  If a particular entry from the reflection list has less numbers than the number of items after `dataformat`, the behavior of the program depends on the type of the missing item. If `intensity`, `amplitude` or "`a`" is missing, program terminates with an error message. If "`b`" or `group` is expected, value 0 is assumed. If `phase` is expected, unknown phase is assumed (like if there were a questionmark).

  If a reflection entry has more numbers than the items after `dataformat`, these additional numbers are silently ignored.

  The default "`a b`" has been chosen for a backward compatibility with the files produced for BAYMEM, for example by JANA2000.

Although all this seems very complicated, the practice is much simpler. Unless you want to use a special format of the data or some special features of SUPERFLIP, you are likely to use either `dataformat amplitude` or `dataformat intensity`. Each reflection entry will then consist of the reflection indices and one number, this number being either $|F|^2$ (for intensity) or $|F|$ (for amplitude).

If is also possible to use a preset for reflection list format. These presets are especially useful when reading reflections from external file (see keyword `fbegin - endf`), but can be useful also for internal reflection list. The presets influence not only the keyword `dataformat`, but also keywords `reflstartline`, `reflendline` and `dataitemwidths`. Currently supported presets are:

- `m91`: equivalent to typing

  ```
  dataformat intensity
  reflendline " 999"
  dataitemwidths 4 9
  ```

- `m90`: equivalent to typing

  ```
  dataformat intensity
  reflstartline "Data Block"
  reflendline "Data Block"
  dataitemwidths 4 9
  ```

- `shelx`: equivalent to typing

  ```
  dataformat intensity
  reflendline "   0   0   0"
  dataitemwidths 4 8
  ```

4.1.11 name: dataitemwidths

- value: list of positive integers
- default: empty list
- description: Defines the width (in characters) of each item in the reflection entry. This keyword is needed only if the reflection list in is a fixed format AND if there is a danger that two items could be concatenated, i.e. there is no space between them. Otherwise there is no need to define this keyword. The first number in the list is the width of the index entry. This number should be given only once and will be used for all indices. Following numbers define the width of items following the reflection indices. If no `dataitemwidths` is provided, it is assumed that the items in the reflection list are separated by one or more spaces.

  Example: Reflection list containing line "`0 0-10 6789.0`" will generate an error under default conditions, because the second and third index are concatenated. The list will, however, be correctly read with "`dataitemwidths 3 9`".

4.1.12 name: delta

- value: positive real number/AUTO [static/dynamic/sigma]
- default: AUTO
- description: Determines the handling of δ, the central parameter of the charge flipping algorithm. There are two basic ways of setting δ: automatically by the program, or directly be the user. With the setting `delta AUTO` the program tries to find the proper δ automatically. This is the recommended option in most cases, because for most of

the structures the automatic procedure works well. The other option is to set delta to a fixed value. This is achieved by setting `delta` *value*.

The original algorithm was developed with δ that is defined at the beginning of the iteration and then kept fixed during the iteration. Wu, Spence, O'Keeffe and Groy (2004) have developed another way of handling δ. In their modification, δ is defined as a fraction of all pixels that are to be flipped. At each iteration, the pixels are sorted according to their electron density, and the pixels with order lower than δN_{pixels} are flipped. Clearly, delta must be now smaller that one, and it is typically around 0.75. This dynamic determination of the flipping threshold has the advantage of being independent of the scale of the structure-factor amplitudes, but otherwise does not seem to outperform the original variant, and the sorting of pixels to find the dynamic threshold represents an additional computational cost.

The absolute value of δ depends on the scale of the data. A parameter k_{ed} defined as $\delta = k_{ed} * \sigma(\rho)$ is, on the other hand, independent of the scale, and provides thus a convenient means to define delta. k_{ed} has typically the value in range $1.0 - 1.3$. If δ is defined by setting `delta` *value* `sigma`, *v*alue is interpreted as k_{ed}.

The original way of handling δ is defined by setting `delta` *value* `static` (synonym `absolute`), the dynamic δ of Wu and Spence is defined by `delta` *value* `dynamic` (synonym `fraction`). Default is static. Automatic search of delta uses the original, static variant.

4.1.13 name: derivesymmetry

- value: yes/no/use [limit agreement factor]
- default: no; default limit agreement factor 25%
- description: If `yes`, SUPERFLIP will try to derive the symmetry of the reconstructed density. Only the cell parameters and the density itself are used for this procedure, so the derivation is completely independent of the symmetry entered in the input file.

  The derivation proceeds in three steps:

  - First all symmetry operations compatible with the lattice are derived and their agreement factors are calculated. Their list is written out in order of increasing agreement factor (*i.e.* the best come first).

  - Then the symmetry operations with agreement factor under limit agreement factor are used to construct a complete space group.

  - Finally, if the dimension of the density is 3 or 3+1, the symbol of the space group is derived.

  Note that with `derivesymmetry yes` the shifting and averaging of the density still remains based on the symmetry from the input file (see keyword `symmetry - endsymmetry`). If you want to use the derived space group for the shifting or averaging, use `derivesymmetry use`.

  Please note also that the detection of symmetry still remains a tricky task and should be by NO MEANS used blindly. Please read Chapter 7, which contains more information about handling symmetry in SUPERFLIP.

4.1.14 name: dimension

- value: positive integer
- default: 3

- description: Defines the dimension of the problem. For standard structures dimension is 3, for modulated structures dimension is $3+d$ where d is the number of independent modulation vectors.

4.1.15 name: expandedlog

- value: yes/no
- default: no
- description: During the whole run of the program there is a lot of information produced that can be of potential interest in some situations, but is uninteresting in most of the cases. `expandedlog` allows to switch between the concise form of the file `jobname.sflog`, which however contains the general information about the iteration, and the expanded form, which provides a detailed record of all steps of the data processing, iteration, and output. For the description of the differences between the two forms of the log-file see Section 5.

4.1.16 name: fastfft

- value: yes/no
- default: no
- description: Normally it is possible to exactly reproduce a run of SUPERFLIP by using a fixed random seed (see keyword `randomseed`). This can be useful, if a solution of a hard-to-converge structure is obtained, and the calculation should be repeated with different parameters. However, this option comes at the cost of the efficiency of the FFT routine (about 20-30% slower). If the option to reproduce the calculation with the same random seed is not needed, then the setting `fastfft yes` can be used to apply the optimal FFT algorithm, and thus save some 10-25% of the computation time.

4.1.17 name: fbegin (– endf)

- value: in the one-line form: filename of the external reflection file

  in the multiline form: each line contains one structure factor in the form:

  ```
  h k l m... reflection information
  ```

  where the reflection information can be one or more of the following: intensity, amplitude, A and B component of the structure factor, phase, group number.
- default: Compulsory keyword – no default, unless `perform symmetry` is used
- description: Information about the input data. Each reflection is defined by its reflection indices (their number must correspond to the value of keyword `dimension`). The reflection information that follows the indices must define the amplitude of the structure factor, and optionally may define its phase. The phase is not used in the actual charge-flipping iteration, but it can be useful under special circumstances (see keywords `perform` and `usephases`). For more information on how the reflection information is handled see keyword `dataformat` and `dataitemwidths`. Note that SUPERFLIP makes no use whatsoever of $\sigma(F)$.

  The external reflection file is handled as if its contents was entered between keywords `fbegin` and `endf`, with one exception - the possible header and footer of the external reflection file can be skipped using keywords `reflstartline` and `reflendline`.

4.1.18 name: filebase

- value: valid filename
- default: the name of the input file without the extension
- description: SUPERFLIP writes the log of the calculation in a file *filebase*`.sflog`. This keyword defines the name of the output file without the extension `.sflog`. Most often this keyword is omitted and the default behavior is used. In the default case the filebase is derived from the input file in the following manner: If the name of the input file does not contain a dot (.), the filebase is equal to the name of the input file. Otherwise the filebase is equal to the name of the input file without the part starting at the last dot. In other words, the filebase is simply the input filename without the extension.

4.1.19 name: finevoxel

- value: no/`AUTO`/real_number `angstrom`/real_number `shannon`/list of positive integers; their number must be equal to dimension
- default: AUTO
- description: Defines the grid for the final noise removal step of the iteration. Using a fine grid sampling leads to better density maps, but takes much more time to calculate, and is not really advantageous during the iteration itself. Therefore SUPERFLIP can resample the density after the convergence, but before the noise removal step (see keyword `polish`). This keyword serves to define the grid spacing for the polishing step.

  Using `finevoxel no` turns off the resampling of the density, and the same grid will be used for polishing as for the iteration. The default setting (`finevoxel AUTO`) is currently equivalent to `finevoxel 0.2 angstrom`. The format, options and restrictions on this keyword are the same as on the keyword `voxel` - please see that keyword for details.

4.1.20 name: fullreflections

- value: valid filename
- default: no list
- description: Occasionaly it can be useful to have a complete list of all reflections (input reflections and their symmetry-equivalents) with phases from charge flipping. If the keyword `fullreflections` is given, SUPERFLIP will write out such a complete list into a separate file with name given after the keyword `fullreflections`. The file contains one line per reflection with format `h k l... A B`.

4.1.21 name: fwhmseparation

- value: positive real number
- default: 0.2
- description: Useful only in connection with powder data with supplied FWHM for each reflection. Defines if a pair of reflections neighboring in the powder diffraction diagram should be considered as overlapped. The criterion for overlap is: $(2\theta_2 - 2\theta_1) < (fwhm_1 + fwhm_2)/2$.

4.1.22 name: histogram (- endhistogram)

- value: in the multiline form: each line contains one entry of the histogram (two real numbers)

  in the one-line form: composition [Biso]/structure filename.cif

- default: no histogram

- description: SUPERFLIP allows for an improvement of the density by a technique called histogram matching (Zhang and Main, 1990). In short, the density values are modified so that they match a predefined histogram, possibly derived from a similar structure or calculated. To calculate a histogram, a reference density must first be available that has the expected histogram. Then the pixels of the reference density must be sorted in ascending order, from smallest to largest. The histogram is defined by pairs of values. The first value gives the rank of a density pixel in a sorted list of density pixels divided by the total number of pixels (*i.e.* the first value is a number between 0 and 1). The second value is the density value for that pixel. The rank values can be equally spaced between 0 and 1, but need not be. They can be more densely spaced in intervals where the histogram changes more, and less densely spaced in intervals with small variation. There are three basic options how to enter histogram in SUPERFLIP:

  – explicitly: The histogram is entered as a list of pairs of numbers as described above. Then the entry for histogram can look something like this:

```
histogram
    0.00000     -3.36059
    0.00084     -2.26701
    0.00129     -2.14628
    0.00205     -2.02972
.........
    0.85162      1.14774
    0.86876      1.26416
    0.88166      1.38058
.........
    0.99868     15.54898
    0.99912     16.23262
    0.99956     17.21292
    1.00000     19.92073
endhistogram
```

  Note that the list must always start with 0.0000 (the minimum value in the reference density), and end with 1.0000 (the maximum value).

  – by supplying a model structure: SUPERFLIP can calculate the histogram from a structure supplied as a cif file. In such case the keyword `histogram` is a one-line keyword with the form `histogram structure filename.cif`. Please note that the parser of the CIF files in SUPERFLIP is very basic and it is the responsibility of the user to supply a valid CIF file compatible with the input file for SUPERFLIP.

  – from composition: In this case Superflip will calculate the histogram only from the expected chemical composition by first generating a random structure with the desired composition and then analyzing this random structure. The form of the keyword is `histogram composition [Biso]`. Composition must be supplied using the appropriate keyword (see keyword `composition`). Biso is the overall temperature factor used for generating the histogram. If omitted, the value from Wilson plot will be used. Note however, that the values from Wilson plot can be biased, especially for powder data. Sometimes a qualified guess gives better values.

4.1.23 name: hmparameters

- value: two integer numbers [yes/no]
- default: 0 0 true; applicable only if keyword `histogram` is present
- description: The histogram-matching procedure (see keyword `histogram - endhistogram`) is usually not performed after every cycle. It is first performed at cycle number *hmstart* and then every cycle for which (cycle-*hmstart*) mod *hmstep* = 0. *hmstart* and *hmstep* are the first and second number following the keyword `hmparameters`, respectively; value 0 for *hmstart* switches off histogram matching. The histogram matching can, but need not be followed by repartitioning of the intensities. If the third argument of `hmparameters` is `yes`, intensities within overlap groups are repartitioned, otherwise only the phases of reflections are updated.

4.1.24 name: lambda

- value: positive real number
- default: required if FWHM is supplied for powder data, otherwise inapplicable
- description: The wavelength at which the powder diagram was recorded. Used to calculate the 2θ angle of a reflection..

4.1.25 name: maxcycles

- value: non-negative integer
- default: 10000
- description: Defines the maximum number of iteration cycles. If this number of cycles is reached, the iteration is interrupted and no convergence is detected. Alternatively, this number can be passed to the program as a second command-line argument (see Section 3.3. The command-line argument has priority over the value in the input file.

4.1.26 name: missing

- value: zero/float/bound/boundsum [resolution_limit [upper_bound]]
- default: float 0.4 for `normalize no`, bound 0.4 4 otherwise
- description: Defines the handling of the missing low-angle reflections. In the standard variant of charge flipping the amplitudes of all the non-measured reflections are set to zero at each iteration. This is appropriate for the high-angle reflections, but it reduces the performance drastically, if strong low-angle reflections are missing. SUPERFLIP offers four modes of handling the missing reflections:
  - `zero`: The amplitudes are set to zero at avery cycle of the iteration - the behavior of the original charge flipping algorithm.
  - `float`: The amplitudes of the missing reflections are unconstrained and are freely evolving as the algorithm progresses. In other words $F_{\mathbf{h}} = G_{\mathbf{h}}$.
  - `bound`: as `float`, but amplitudes that exceed upper_bound times expected value are reset to upper_bound times expected value. Expected value is equal to one, if the reflections are normalized, otherwise it is calculated from the Wilson plot.
  - `boundsum`: The sum of all amplitudes of the missing reflections is constrained to the sum of expected amplitudes. This option should be used only if the set of missing reflections is likely to contain both weak and strong reflections.

The second argument after the keyword defining the handling of the missing reflections is the maximum resolution (as $\sin\theta/\lambda_{\max}$ or $d_{\min}$ depending on the keyword `resunits`) up to which the reflections are added to the missing list. Default is $\sin\theta/\lambda_{\max} = 0.4$, or $d_{\min} = 1.25$. The third, optional argument is the upper bound used for the "`bound`" mode. Which option is the most efficient remains to be established, but the simple `float` option performs better than the `zero` option, while it does not require the Wilson plot. It is thus the most general option. The option `bound` performs distinctly better that the option `float`, but the optimum range of the parameter upper_bound has not yet been established. For normalized reflections it seems to be between 3 and 5, for unnormalized data values around 10 seem to give good results, but these numbers are based on a limited number of tests and should not be taken for granted.

4.1.27 name: modelfile

- value: valid filename of an existing file [average phase deviation]
- default: no model file; compulsory only for setting `perform symmetry`
- description: Contains the name of the model density file. The model file can be used in two ways. In connection with the setting `perform symmetry` it contains the density to be tested for symmetry (for more information see keyword `perform`). Otherwise the model file will be used for calculation of the initial phases instead of assigning the initial phases at random. With this option Superflip can be used to complete a partial structure model.

  The model structure can be supplied either as a density in jana, xplor or ccp4 format (see the keyword `outputformat`), or as a CIF file with the structure information about the model structure. In the latter case the model density is calculated from the structure in CIF.

4.1.28 name: modelformat

- value: jana/xplor/ccp4/cif
- default: the format is guessed from the extension of the model file
- description: The format of the model density (see the keyword `modelfile`). For more information about the formats see the keyword `outputformat`.

4.1.29 name: normalize

- value: wilson/local/no
- default: no
- description: `normalize wilson` (synonym `normalize yes`) will lead to the normalization of the input amplitudes (replacing F's by E's). The chemical composition of the structure must be given by keyword `composition`. Alternatively the "local normalization" can be used with `normalize local`. This normalization is based on the program ECALC by Ian Tickle from the CCP4 program suite (CCP4, 1994). It does not make use of the chemical composition, but requires a sufficiently large number of reflections in each resolution shell, and is thus applicable especially for large structures and macromolecules. The number of resolution shells is by default determined automatically, but can be set by user using the keyword `nresshells`.

4.1.30 name: nresshells

- value: positive integer
- default: 100, or such that the number of reflection in each shell is at least 200.
- description: Defines the number of resolution shells used for normalization using the "local" normalization procedure (see keyword `normalize`).

4.1.31 name: outputfile

- value: valid filename [another valid filename...]
- default: compulsory keyword – no default
- description: Defines the name of the file holding the resulting electron density. More than one filename can be given, maximum is 10. SUPERFLIP will save the density in all the files in formats given either by the keyword `outputformat`, or guessed from the file extension. See keyword `rewriteoutput` for more information about handling the output files.

4.1.32 name: outputformat

- value: jana/xplor/ccp4/m80 [formats of other output files according to the keyword `outputfile`
- default: guess the density format from the extension of the of the output filename
- description: Defines the format of the output electron density. The setting `outputformat jana` results in saving the density in the format of the crystallographic system Jana2000 (Petříček, Dušek and Palatinus, 2000). The standard extension is `m81`. It is a binary format suitable for saving electron density in arbitrary dimensions up to six. It can be viewed in Jana2000/Jana2006 in a form of contour plots in arbitrary sections or projections. Setting `outputformat ccp4` will produce a ccp4-compliant format - a binary format standard in macromolecular crystallography. Can be read by the program UCSF Chimera (`http://www.cgl.ucsf.edu/chimera/`), and other programs from the macromolecular field. Setting `outputformat xplor` results in an ASCII output format of Xplor/CNS (extension `xplor`) – a program system for computational structural biology (`http://cns-online.org/v1.2/`). It can be viewed by the UCSF Chimera and also by the program VESTA (`http://www.geocities.jp/kmo_mma/crystal/en/vesta.html`). Being an ASCII format, it is easy to modify it to any other format. The native xplor format is suitable only for 3D data. SUPERFLIP uses a generalized xplor format for saving electron densities with more than three dimensions. See section 5.1.1 for closer description of the format. Setting `outputformat m80` will produce a list of reflections with phases that can be used in Jana2000 to produce arbitrary section through the density. Standard extension is `m80`. See section 5.1.4 for closer description of the format. If `searchsymmetry` is set to `average`, only reflections present in the input file are listed. In all other cases the list contains complete set of reflections.

  Other formats of the output file can be added upon request, provided a transparent description of the format (or a reference thereto) is supplied together with the request.

4.1.33 name: perform

- value: CF/lde/general/fourier/symmetry
- default: CF

- description: Apart from performing the density reconstruction by charge flipping (setting `perform CF`), SUPERFLIP can perform the low-density elimination method (setting `perform lde`, and also do a simple Fourier transform of the input data (setting `perform fourier`), provided the phases of the structure factors are supplied (see keyword `fbegin - endf`). Setting `perform symmetry` can be used if you have a density file from the previous calculation (or from other source than SUPERFLIP) and want to check the symmetry of that density and/or shift and average the density according to the symmetry.

  The low-density elimination method is closely related to charge flipping, the only difference is in the density modification step, where instead of flipping all charge below zero is set to zero. For more information consult Shiono and Woolfson (1992).

  In reality both the charge-flipping algorithm and the low-density elimination are just special cases of a general iterative algorithm, of which other flavors are known as Fienup's hybrid input-output algorithm or Elser's Difference Map. For the discussion of the relationship between the algorithms see section 2. The setting `general` allows an application of the general algorithm. This algorithm has six parameters, and all of them must be listed after the word `general`, in order (see section 2) $\beta_1, \gamma_{1R}, \gamma_{1D}, \beta_2, \gamma_{2D}, \gamma_{2R}$.

  Performing simple Fourier transform on the input data can be occasionally useful for checking the internal consistency of the data, if the structure is already known and SUPERFLIP is used to test the behavior of charge flipping on that data set.

  Sometimes it happens that a density is available and the only problem is to test it for the presence of the symmetry operations and shift or average it over symmetry. For this purpose the setting `perform symmetry` has been introduced. This setting causes the program to read in a user-supplied scattering density (via keyword `modelfile`), then skip completely the charge flipping iteration and proceed directly to handling of the symmetry. The keywords governing the charge flipping iteration are irrelevant in this case, but all the keywords influencing the origin shift, averaging and symmetry remain fully functional. The keywords that are used are: `symmetry - endsymmetry`, `centers - endcenters`, `searchsymmetry`, `derivesymmetry` and `referencefile`. Apart from these keywords, only the compulsory keywords are required in the input file, namely `modelfile`, `outputfile`, `voxel`, and `cell`. The reflection list `fbegin - endf` is not required. If present, the structure factors of reflections present in the list are calculated from the tested density. Otherwise structure factors of all possible reflections within the grid are calculated and used in the tests.

4.1.34 name: polish

- value: yes/no [integer number of cycles]
- default: yes
- description: The settings of the algorithm that are the best to achieve the convergence are not necessarily the best to obtain a clear map with suppressed noise. With setting `polish yes` the program will perform additional iteration cycles after the convergence has been detected or the maximum number of cycles was reached. It is assumed that the calculation is converged at that moment and these additional cycles are used to "polish" the density to obtain as noise-free result as possible. To achieve this, the parameters during these polishing cycles are changed: the method is changed to low-density elimination and the weakratio is set to zero. Moreover, a finer sampling of the density is used for the polishing step, unless suppressed (see keyword `finevoxel`).

  The default number of polishing cycles is 5, user can change this number by placing an integer number of cycles after `polish yes`.

4.1.35 name: presentationmode

- value: limit delay [limit2 delay2 [limit3 delay3 [...]]]
- default: 0 0
- description: Using this keyword it is possible to slow down the iteration by introducing a delay after every cycle up to the limit. It is possible to introduce an arbitrary number of limits with different delays. The delay is given in ms. As an example, setting `presentationmode 100 500 1000 0 1100 1000` means that the first 100 cycles will be delayed 500ms each, then all cycles up to cycle number 1000 will run without any delay, and cycles 1001-1100 will have a delay of 1 second. Every cycle after the last limit runs without any delay.

  This keyword is useful if you want to demonstrate the different stages of the iteration, for example in a lecture, and the iteration is too fast. It is particularly useful for connection with the keyword `viewprogress`.

4.1.36 name: qvectors – endqvectors

- value: each line contains coordinates of one q-vector as decimal numbers
- default: compulsory keyword if dimension > realdimension, otherwise inapplicable
- description: The list holds the definition of the q-vectors in a modulated structure. The number of q-vectors must be equal to the difference between dimension and realdimension (see the corresponding keywords). The number of components of each q-vector is equal to realdimension.

4.1.37 name: randomseed

- value: AUTO/non-negative integer
- default: AUTO
- description: Defines the seed for initialization of the random number generator. Setting `randomseed AUTO` causes the program to generate the seed from the system time. Positive integer is taken as a random seed without any modification. In this way it is possible to exactly reproduce one calculation several times.

4.1.38 name: realdimension

- value: positive integer
- default: 3
- description: Defines the dimension of the real space. For normal crystal structures this is always 3. Other values might be used in case of data from the two-dimensional surface scattering, calculation of quasicrystal densities, or theoretical experiments with higher-dimensional crystallography.

4.1.39 name: referencefile

- value: a valid filename of an existing file
- default: no reference file

- description: Occasionally the result of the charge flipping needs to be compared with an electron density obtained otherwise, or with another result of charge flipping on the same data. The location of the origin of the space group in the electron density is not sufficient for bringing the density itself always to the same position. This has two reasons. First, some symmetry operations generate equivalent symmetry operations elsewhere in the cell, if combined with the lattice translation (a simple example: a mirror plane in the origin of an ortorhombic cell generates another mirror plane at 1/2). The second reason is that many non-centrosymmetric space groups have one or more directions that cannot be fixed by the symmetry at all and the resulting density, although averaged over symmetry, will be randomly shifted along these directions.

  For these reasons the user has the possibility to supply a reference electron density. Superflip will align the resulting electron density with the reference density, allowing for a direct comparison of them. Another application of this technique is the possibility of summing up several results of charge flipping, which should result in cleaner density map, since the signal sums up, but the random noise cancels upon the summation.

  The reference structure can be supplied either as a density in jana, xplor or ccp4 format (see the keyword `outputformat`), or as a CIF file with the structure information about the reference structure. In the latter case the model density is calculated from the structure in CIF.

4.1.40 name: referenceformat

- value: jana/xplor/ccp4/cif
- default: the format is guessed from the extension of the reference file
- description: The format of the reference density file (see the keyword `referencefile`). For more information about the formats see the keyword `outputformat`.

4.1.41 name: reflectionlist

- value: general/unique
- default: general
- description: `reflectionlist general` specifies a general list of reflections that can contain multiple instances of the same reflections as well as symmetry-equivalent reflections. Reflections are sorted and averaged in the program. `reflectionlist unique` indicates that the reflection list is already merged and contains only unique reflections (for the space group specified in the input file). For this setting the sorting, averaging and merging of reflections is skipped, thus saving considerable time for huge data sets.

4.1.42 name: reflendline

- value: any character string, possibly included in quotas (")
- default: no reflendline
- description: Line following the last reflection entry in the reflection file. Useful only if the reflections are read from an external reflection file (see keyword `fbegin`). All lines of the reflection file starting from the reflendline are ignored. If reflendline contains spaces at the beginning, the whole line must be included in quotas to explicitly include the leading spaces.

4.1.43 name: reflstartline

- value: any character string, possibly included in quotas (")
- default: no reflstartline
- description: Line immediately preceding the first reflection entry in the reflection file. Useful only if the reflections are read from an external reflection file (see keyword `fbegin`). All lines of the reflection file up to reflstartline are skipped. If reflstartline contains spaces at the beginning, the whole line must be included in quotas to explicitly include the leading spaces.

4.1.44 name: repeatmode

- value: never/nosuccess/always/*integer number of repetitions* [sumall/sumgood]
- default: never
- description: SUPERFLIP can repeat the whole calculation. With setting `repeatmode never` (which is the default) the program will stop after one calculation, i.e. when the calculation converges or if the maximum number of cycles is reached (see keyword `maxcycles`). With the setting `repeatmode nosuccess` the calculation is repeated until the convergence is detected. `repeatmode always` will cause the program to repeat the calculation indefinitely. The saving of the best density (or several best densities) is controlled by the keyword `bestdensities`. If integer number n is given as the first value, the calculation is repeated n times.

  If the optional keyword `sumall` or `sumgood` is present, then the densities from individual runs are summed up, and at the end the averaged density is written out to the file(s) given by the keyword `outputfile`. For `sumgood` only the results of converged runs are summed up, for `sumall` all densities are summed up irrespectively of the convergence status. Before summing up the individual densities are processed according to all settings of the program, including the settings influencing the handling of the symmetry. If external reference density is provided (see keyword `referencefile`), all densities are aligned to the reference density before summing, otherwise the summed density itself is used as a reference density for aligning the subsequent densities.

4.1.45 name: reslimit

- value: high-resolution-limit [low-resolution-limit]
- default: no reflections excluded
- description: This keyword allows to exclude the high- and low-resolution shells from the reflection list. The resolution limits can be given either as d-spacing, or as $\sin\theta/\lambda$. The units are selected with the keyword `resunits`. If only one number is given, then it is interpreted as the high-resolution limit ($d_{\min}$ for d and $(\sin\theta/\lambda)_{\max}$ for $\sin\theta/\lambda$). If two numbers are present, the high- and low-resolution limits are recognized automatically.

4.1.46 name: resunits

- value: `d/sthl`
- default: `sthl`
- description: Selects the units used to define resolution limits in other keywords (`missing` and `reslimit`).

4.1.47 name: rewriteoutput

- value: yes/no
- default: yes
- description: With `rewriteoutput yes` both the log file and the density file are overwritten by the new log file and density file. With `rewriteoutput no` the log file is appended and the density file is not overwritten, if it exists. Instead of that the density is written to a file `sfrhoXX.`*ext*, where `XX` is a number between 00 and 99, and *ext* is a format-specific extension, namely `.m81` for the jana format, `xplor` for the xplor format, and `ccp4` for the ccp4 format. First unused number is selected. If all files `sfrho00.`*ext* through `sfrho99.`*ext* exist, an error message is written to the log file and the program terminates without writing the density. The information about the name of the density file is always written at the end of the log file.

4.1.48 name: searchsymmetry

- value: no/shift/average
- default: average
- description: The electron density is always reconstructed in $P1$, i.e. without the use of the symmetry. But of course, the resulting density still (approximately) obeys the underlying symmetry of the structure, only the origin of the space group is randomly shifted in the cell. Thus, the symmetry elements can be localized in the resulting electron density. This keyword defines, how the program should handle the symmetry. `searchsymmetry no` prevents any search for the position of the symmetry elements. `searchsymmetry shift` leads to the location of the origin of the space group in the density and subsequent shifting of the density. However, no averaging is performed, the density is only shifted. `searchsymmetry average` leads to the location of the origin, shifting of the density and averaging of the density over the symmetrically equivalent pixels, so that the resulting density has exactly the symmetry of the space group. The last setting is recommended, unless problems occur with the location of the origin.

4.1.49 name: skipstartcycles

- value: positive integer
- default: 0
- description: At the beginning of the iteration a transition period can appear that is confusingly similar to a convergence. To prevent the program from detecting this false convergence this keyword can be used to force the program to ignore a number of starting iteration cycles, and start analyzing the convergence criteria later.

4.1.50 name: symmetry – endsymmetry

- value: in the multiline form: each line contains one symmetry operation in the one-line form

  in the one-line form ccp4:*ccp4 code of the space group*
- default: compulsory keyword – no default
- description: Defines the symmetry of the structure. Its primary use is to average the reflections and expand them to full sphere. It need not be known for the charge-flipping iteration, but it can be used to recover the symmetry of the resulting density

(see keyword `searchsymmetry`. If the list of reflections is sufficiently complete and the space group is uncertain, space group $P1$ can be always used.

The symmetry operations are given in a one-line form known from the International Tables for Crystallography. SUPERFLIP accepts both the notations x, y, z, and $x1$, $x2$, $x3$. For higher-dimensional cases letters x, y, z, t, u, v can be used, but the notation $x_1 \cdots x_n$ is preferable. The translational part can be given both as a fraction and as a decimal number. A complete space group must always be listed, including the identity operation. The centering vectors are, however, listed separately (see keyword `centers - endcenters`). Two examples will illustrate the form of the input:

$P4_2/n$:

```
symmetry
    x     y     z
1/2-x 1/2-y     z
   -y 1/2+x 1/2+z
1/2+y    -x 1/2+z
   -x    -y    -z
1/2+x 1/2+y    -z
    y 1/2-x 1/2-z
1/2-y     x 1/2-z
endsymmetry
```

(3+1)-dimensional superspace group $Cmcm(0\beta0)s0s$:

```
symmetry
 x1  x2     x3     x4
-x1  x2     x3 1/2+x4
 x1 -x2 1/2+x3     x4
 x1  x2    -x3 1/2+x4
-x1 -x2    -x3    -x4
 x1 -x2    -x3 1/2-x4
-x1  x2 1/2-x3    -x4
-x1 -x2     x3 1/2-x4
endsymmetry

centers
1/2 1/2 0 0
endcenters
```

For convenience the symmetry can be also given using the code used in the CCP4 suite (CCP4, 1994). For the standard settings of the space groups the ccp4 code corresponds to the number of the space group in *International Tables for Crystallography, Vol. A*. Additional codes are available for some alternative settings of some space groups. If you want to use this option, please refer to the documentation of CCP4, especially to the description of the symmetry library syminfo.lib. For example, the space group $Pnma$ can be entered as `symmetry ccp4:62`.

4.1.51 name: terminal

- value: yes/no [keep]
- default: yes

- description: If yes, a short information about the progress of the calculation is written on the standard output. This output can be suppressed by setting `terminal no`. Such setting can be useful if you run SUPERFLIP in a batch mode on a remote computer, in which case you do not want to bind the execution of the program to an existence of specific terminal window. If a word "keep" is present as the second word after the keyword, the program will ask for pressing Enter before quitting. That is useful if the terminal window closes automatically after the end of the execution, and the user wishes to see the output before closing the window.

4.1.52 name: testsymmetry – endtestsymmetry

- value: each line contains one symmetry operation in a one-line form
- default: no test symmetry operations
- description: Occasionally the symmetry is not known with certainty and several space groups are possible. SUPERFLIP can take the list of symmetry operations, and test, how well is the corresponding symmetry present in the density reconstructed by charge flipping. In combination with `searchsymmetry no` the whole density is searched for the optimal position of each of the listed symmetry operations, with other settings only the positions allowed by the ambiguity of the position of the origin are tested (compare keyword `referencedensity`). The syntax of the symmetry operations is the same as in the keyword `symmetry - endsymmetry`.

  NOTE: This option is obsolete and will not be supported in the future. It is largely replaced and improved by the keyword `derivesymmetry`.

4.1.53 name: title

- value: string of characters up to 132 characters long
- default: no title
- description: Title of the calculation. Is written in the log file and can serve for the identification of the job. The program makes no other use of this string.

4.1.54 name: usephases

- value: no/firstcycle/always/integer number
- default: no
- description: With `usephases firstcycle` the starting phases are not chosen randomly, but the phases present in the input file are used. If only some phases are known and some not (see keyword `dataformat` for details), the unknown phases are taken at random. With `usephases always` the known phases are reset to the input values in each iteration cycle. If a number n is given as a value to `usephases`, the phases are set to their input values only in the first n cycles of the iteration and after that they are allowed to change freely.

4.1.55 name: viewprogress

- value: no/chimera [period]
- default: no

- description: Using this keyword it is possible to call the program UCSF CHIMERA (`http://www.cgl.ucsf.edu/chimera`) to visualize the progress of the iteration. SUPERFLIP calls CHIMERA with a script that makes CHIMERA read the current density written out by SUPERFLIP, and then reread the file in regular intervals. Period refers to the number of cycles elapsed before the next update of the density file by SUPERFLIP. The default value is 5, and causes SUPERFLIP to write out a density for CHIMERA every five cycles. Clearly, this option will slow down the calculation, because of the time needed to write out the density, and because of slowing down the computer by simultaneously running SUPERFLIP and CHIMERA (unless you use a multiprocessor or multicore machine). However, the slowing down is not dramatic (factor of two or so for a typical calculation).

  To enable this option, you only have to download and install UCSF CHIMERA (installation packages are available for Windows, MacOS X and Linux), and add the path to the CHIMERA executable to your environment variable PATH.

  Under Windows a message "Access is denied" occasionally appears in the SUPERFLIP window. Apart of being a bit annoying it does not do any harm. Its frequency can be decreased by increasing the value of `period`.

  A note for experts: SUPERFLIP writes out a default python script for CHIMERA before calling the program. However, you can supply your own python script that will do other things, for example record a movie of the iteration. The python script must be located in the same directory as the input file for SUPERFLIP, and it must have the name `watchsuperflip.py`.

4.1.56 name: voxel

- value: `AUTO`/real_number `angstrom`/real_number `shannon`/list of positive integers; their number must be equal to dimension
- default: AUTO
- description: Defines the grid on which the density is computed. It can be defined either as a list of numbers, each representing the grid division along one unit-cell edge, as a size of a pixel, or as a Shannon rate. If the grid division is given explicitly, then the numbers are subject to the following restrictions:
  - The grid division in each dimension must be larger that two times the largest reflection index in that dimension.
  - The grid must be compatible with the symmetry, i.e. each grid point must be mirrored by all symmetry operations onto itself or onto another grid point. For example, a spacegroup containing a 6_1 axis along z must have the division along the third axis that is a multiple of 6.
  - The last restriction is not a must, but it is recommended that the grid divisions can be factorized into small primes, preferably 2 and 3. This speeds up the fast Fourier transforms, that take most of the time in the calculation.

  With the setting `voxel AUTO` Superflip will automatically calculate grid division compatible with the above criteria. The number of voxel in each dimension i will be set as small as possible, but always larger than $h_{max}(i) * 2 + 2$.

  If the size of a grid point is given (by one fractional number, in Ångstrom), then SUPERFLIP will calculate automatically such a grid spacing that can adopt all the reflections and is compatible with the symmetry (as for `voxel AUTO`), and make sure that the grid-point size in all directions is not smaller than the specified number. For example, `voxel 0.2 angstrom` will lead to a grid division with voxel size *at most* $0.2 \times 0.2 \times 0.2$Å^3.

Shannon rate specification gives the ratio between the number of voxels along each direction, and the maximum index along this direction. The minimum Shannon rate that allows adopting all reflections is 2. Increasing the Shannon rate results in a finer sampling.

4.1.57 name: weakratio

- value: positive real number [positive integer] [`relative/absolute`]
- default: 0.0 1 absolute
- description: In the second article of the authors of charge flipping (Oszlányi and Sütő, 2005) it has been shown that the convergence can be substantially improved by perturbing not only the electron density in the direct space, but also the phases of the structure factors in reciprocal space. The perturbation is achieved by shifting the phases of certain portion of the weakest reflections by $\pi/2$ in each iteration cycle and by not replacing the calculated amplitudes with the observed ones. The keyword `weakratio` serves for defining the fraction of the reflections that are considered weak and subject to the phase shift. Typically the value of `weakratio` is between 0.2 and 0.3. However, note that the use of `weakratio` yields somewhat more noisy and less accurate maps due to the loss of information from the weak reflections. But even these maps usually represent a pretty good approximation of the real density. Under special circumstances it can be desirable to postpone the application of the perturbation to later stages of the iteration. The second argument specifies the iteration cycle from which on the weak-reflection perturbation is applied. For special applications it is also possible to define the limit for weak reflections not as a fraction of all reflections, but as an absolute value of the amplitude. This can be achieved by using the syntax `weakratio` *`threshold starting cycle`* `absolute`. This saves some time for sorting the reflections. Note however that the threshold applies to amplitudes after all corrections (for the Debye-Waler factor, normalization etc.).

4.2 Examples of an input file

4.2.1 A minimalistic example

This example contains a minimal set of instructions. Despite of its simplicity it is likely to work for most simple structures.

```
title A minimalistic input file
cell 5.2 5.8 4.4 90. 90. 98.
outputfile example1.m81

#Space group P2/m
symmetry
 x1     x2      x3
-x1    -x2      x3
-x1    -x2     -x3
 x1     x2     -x3
endsymmetry

fbegin
   7   0 -3     5.7445626
   7   1 -3     5.3385391
   8   1 -3     7.0710678
    .
    .
    .
endf
```

4.2.2 A realistic example for a modulated structure

This is a real example of a file that illustrates many options available in SUPERFLIP. This file was used to produce the log-file that is described in Section 5.2. This file is included as a sample file in the distribution package of SUPERFLIP.

```
title Cr2P2O7 - incommensurate phase, room temperature
perform CF

# Keywords influencing the form of the files
outputfile Cr2P2O7_sf.m81
outputformat jana
expandedlog no
coverage yes
referencefile Cr2P2O7_ref.m81
referenceformat jana

# Basic crystallographic information
dimension  4
voxel   36  48  24  16
cell     7.0192    8.4063    4.6264   90.00  108.61   90.00
qvectors
  -0.361   0.000   0.471
endqvectors
centers
  0.0  0.0  0.0  0.0
  0.5  0.5  0.0  0.0
endcenters
symmetry
     x1     x2     x3     x4
    -x1     x2    -x3 1/2-x4
    -x1    -x2    -x3    -x4
     x1    -x2     x3 1/2+x4
endsymmetry
testsymmetry
     x1    -x2     x3     x4
endtestsymmetry

# Keywords influencing the algorithm
delta AUTO
weakratio 0.000
biso   0.000
randomseed AUTO
searchsymmetry average
derivesymmetry yes #Check, if the assumed symmetry is indeed present in the density

# List of reflections
dataformat amplitude
fbegin
   0   0   0   2    29.6141853
   0   0   0   4     5.9497900
  -2   0   0  -4     6.4498062
  -2   0   0  -2     7.9498429
   .
   .
   .
endf
```

Chapter 5

Description of the output

There are two main output files. The principal output of the calculation is the file with the electron density. The second output file is the log-file containing the information about the data processing, iteration itself and the results of the calculation.

5.1 The electron density file

The name of the output density file is defined by the keyword `outputfile`, the format is defined by the keyword `outputformat`. Currently SUPERFLIP supports four density formats: the xplor format (defined by `outputformat xplor`), the format ccp4 used by the ccp4 suite (`outputformat ccp4`), the format m81 of the crystallographic package Jana2000/Jana2006 (`outputformat jana`), and the format m80 of Jana2000/Jana2006, which contains a list of structure factors (`outputformat m80`). These structure factors can be used directly by Jana2000/Jana2006 to calculate arbitrary sections through the density. Other formats can be added upon request, provided a transparent description of the format (or a reference thereto) is supplied together with the request.

5.1.1 format xplor

This is an ASCII format of the software package Xplor for structural biology. It can be read and displayed e.g. by the 3D plotting program Chimera. The format assumes the following form:

```
empty line
number of lines of the title
line 1 of the title
line 2 of the title etc. up to the number given on line 2
pixel division: number of pixels along x, first pixel, last pixel; ditto for y and z
cell parameters
ZYX string defining the order of axes from the most slowly to the most quickly varying
     0 - the number of the current layer along z
six density values
six density values
nx.ny values, nx and ny is the number of pixels along x and y. x varying first, y second.
    1 - number of the next layer along z
six density values
    .
    .
    .
-9999 always present at end of the density values
two real numbers for the average and sigma(average). Sigma is not computed by Superflip.
```

For the exact format (exact length and position of the numbers) please check the documentation of Xplor or consult a file in xplor format produced by SUPERFLIP.

5.1.2 Format ccp4

This is the format used by the CCP4 project - Software for Macromolecular X-Ray Crystallography (`http://www.ccp4.ac.uk`). Its standard extension is ccp4. It is a binary format, and its detailed description can be found at `http://www.ccp4.ac.uk/html/maplib.html`.

5.1.3 Format jana

This is the format of the crystallographic software package Jana2000 (Petříček et al., 2000). Its standard extension is .m81. It stores the electron density in single-precision direct-access binary format. It is suitable for storing electron densities up to 6 dimensions and can be viewed in Jana2000. It is beyond the scope of this manual to fully describe this format. The reader interested in the details of the format should consult the source code of SUPERFLIP or Jana2000, or contact the authors of either of the programs.

5.1.4 Format m80

This is the format of the input file for the Fourier module of Jana2000 (Petříček et al., 2000). Its standard extension is .m80. Each line of the file has format (di4,i4,13e12.5), where d is the number of reflection indices. The information in each line is:

reflection indices, number of structure(always 1 in superflip),F_{obs}, F_{obs}, F_{calc}, A, B

The rest of the line is compulsory in the format but it is irrelevant for the output from superflip and is padded with zeroes.

5.2 The log-file

The log-file contains all information about the run of SUPERFLIP. Its name is *filebase*`.sflog`. The filebase is the name of the input file without the extension, unless it is explicitly redefined (see keyword `filebase`).

Depending on the value of the keyword `expandedlog`, the log-file can have two forms: the short, concise record of the main elements of the interation, and the long form, that should be necessary only for diagnostic purposes and in case of the difficulties with some part of the structure solution.

5.2.1 The concise form of the log file

This form is obtained by default or by setting `expandedlog no`. The form of the log-file will be illustrated on the example of a modulated structure of chromium(II)-diphosphate in order to include also the parts specific to the modulated structures. The input file is described in section 4.2.2.

In the following subsections the individual parts of the file will be described.

5.2.2 Information about the input

The first part contains an information about all the control keywords read from the input file. This part is written to be self-explanatory and can be used in any later time to reconstruct the conditions under which the calculation has been performed.

```
-----------------------------------------------
Start of the calculation: 16.JAN 2006, 12:51:24
-----------------------------------------------

###################################################################
# Following data were read from the input file or set as default: #
###################################################################

Job title: Cr2P2O7 - incommensurate phase, room temperature

------------------------
```

```
Information about files:
------------------------
Name of the input file: Cr2P2O7.inflip
Density will be writen in jana format to file   Cr2P2O7_sf.m81
Warning: If the outputfile exists, it will be overwritten by the new density.
Logfile will contain only basic information about the calculation.
Logfile will contain information about the data coverage.

-----------------------------
Crystallographic information:
-----------------------------
Superspace dimension:             4
Dimension of the physical space:  3
Direct cell parameters:          7.0192   8.4063   4.6264  90.0000 108.6100  90.0000  Volume: 258.7095
Reciprocal cell parameters:      0.1503   0.1190   0.2281  90.0000  71.3900  90.0000  Volume:   0.0039
Q-vectors:
  -0.3610   0.0000   0.4710

4 symmetry operations found, their list follows:

   1:  1  0  0  0   0.0000   2: -1  0  0  0   0.0000   3: -1  0  0  0   0.0000
       0  1  0  0   0.0000       0  1  0  0   0.0000       0 -1  0  0   0.0000
       0  0  1  0   0.0000       0  0 -1  0   0.0000       0  0 -1  0   0.0000
       0  0  0  1   0.0000       0  0  0 -1   0.5000       0  0  0 -1   0.0000

   4:  1  0  0  0   0.0000
       0 -1  0  0   0.0000
       0  0  1  0   0.0000
       0  0  0  1   0.5000

The structure is centrosymmetric.
The symmetry operations are to be combined with the following centering vectors:
   0.0000   0.0000   0.0000   0.0000
   0.5000   0.5000   0.0000   0.0000


--------------------------
Settings of the algorithm:
--------------------------
Number of voxels:                36    48    24    16   Total:    663552
The density modification method will be charge flipping.
Delta will be determined automatically.
The iteration will be stopped when the convergence is detected or after 10000 cycles.
The random number generator will be initialized automatically.

The random initial phases will be assigned to the structure factors.
Isotropic Debye-Waller factor:   0.000
Proportion of reflections to be treated as weak:   0.000

The resulting density will be shifted and averaged according to the symmetry operations given above.
Following symmetry operations will be used to locate the origin of symmetry:   4   2
```

5.2.3 Information about the processing of the reflections

The concise form contains only the summary of the reflection-import, i.e. the number of reflections, the maximum indices in the expanded set (can be used to fine-tune the grid size), and (by default or with `coverage yes`) the coverage as a function of the resolution. Always check the coverage at the first run of a new data set to make sure it is sufficient. Low coverage will prevent convergence!

Note: The calculation of the coverage does not work properly for composite structures. This is because of the difficulties with the definition of what is a satellite in such structures (in other words, the W matrix of the composite is not available to the program). However, this problem concerns only the calculation of coverage and has no impact on the calculation itself. The coverage does not work well also for quasicrystals. This might change in the future versions of SUPERFLIP.

```
#################################
# Information about reflections: #
#################################

Number of reflections in the input file:   2409
```

```
-------------------------
Averaging of reflections:
-------------------------
Redundancy   : 2.721
Rint         :  2.540

Maximum indices in expanded reflection set:  10  10   7   4

Coverage statistics of the expanded reflections by shells:
Resolution (sin(th)/l):    0.050    0.100    0.150    0.200    0.250    0.300    0.350    0.400
Resolution (d_min):       10.000    5.000    3.333    2.500    2.000    1.667    1.429    1.250
Obs. refl. in shell:           0       11       53      108      151      236      337      431
Total refl. in shell:          1       15       53      108      151      236      337      431
Coverage in shell:          0.0%    73.3%   100.0%   100.0%   100.0%   100.0%   100.0%   100.0%
Commulative coverage:       0.0%    68.8%    92.8%    97.2%    98.5%    99.1%    99.4%    99.6%

Resolution (sin(th)/l):    0.450    0.500    0.550    0.600    0.625
Resolution (d_min):        1.111    1.000    0.909    0.833    0.800
Obs. refl. in shell:         557      710      828      959      631
Total refl. in shell:        557      710      828      959      631
Coverage in shell:        100.0%   100.0%   100.0%   100.0%   100.0%
Commulative coverage:      99.7%    99.8%    99.9%    99.9%    99.9%
```

5.2.4 Information about the iteration

This part deserves special attention and therefore the listing will be commented by comments interleaved in the sample text:

```
#############
# Iteration #
#############
```

An estimate of delta is written out. The estimate is based on the statistics of the intensities.

```
Estimated delta: 28.1478
```

The random seed used by the program is indicated. This can serve for reproducing the calculation at a later stage.

```
Random seed:  92257248
```

The next part is a record of the search for delta. This is present only for setting `delta AUTO`. The criterion for δ is 0.8 < total charge/flipped charge < 1.0.

```
Searching for a proper delta:
Current delta =   28.14776
   10  R: 50.257  Charge:    2043.65   Peaks:  1.37
Total/flipped ratio =    0.715. Decreasing delta.

Current delta =   25.33298
   10  R: 49.462  Charge:    2190.24   Peaks:  1.40
Total/flipped ratio =    0.870.
Criterion for delta fulfilled, continuing iteration.
```

In the concise form the status is written out only at some cycles of the iteration, not all. Every tenth cycle is listed up to 100 cycles, every hundredth cycle from 100 to 1000 cycles, and every thousandth cycle from 1000 cycles upwards. The header for the iteration record gives the short explanation of the items that occur in each line. The total charge is simply $\sum \rho_i$ summed over all pixels (i.e. $F(\mathbf{0})$). Flipped charge is $\sum |\rho_i|$ summed over all pixels for which $\rho_i < \delta$. Peaks denote the peakiness (skewness) of the density defined as $\gamma = \frac{1}{A} \sum_{i=1}^{N_{pix}} (\rho_i - \bar{\rho})^3$. Score is an overall convergence score based on analysis of several criteria.

```
   20  R: 46.639  Charge:    1576.61   Peaks:  2.41   Score:    ---
   30  R: 38.207  Charge:    1433.81   Peaks:  2.80   Score:    ---
   40  R: 31.340  Charge:    1328.05   Peaks:  2.96   Score:    ---
   50  R: 29.204  Charge:    1305.89   Peaks:  2.92   Score:    ---
   60  R: 29.288  Charge:    1361.29   Peaks:  2.84   Score:    ---
   70  R: 28.799  Charge:    1328.29   Peaks:  2.85   Score:    ---
   80  R: 28.730  Charge:    1333.52   Peaks:  2.84   Score:    ---
   90  R: 28.567  Charge:    1342.82   Peaks:  2.83   Score:    ---
  100  R: 28.009  Charge:    1331.36   Peaks:  2.82   Score:    ---
Calculation successfully converged after   125 cycles.
Last iteration record:
  125  R: 27.505  Charge:    1364.02   Peaks:  2.85   Score:   5.43
```

By default a few cycles of noise removal follow after the convergence:

```
    5 cycles of noise suppression follow:
   5  R: 21.501  Charge:    1082.02   Peaks:  4.46
```

If `derivesymmetry` is set to `yes`, SUPERFLIP derives the space group from the density. You can find more detailed information in Chapter 7.

```
####################################
# Checking the density for symmetry #
####################################

Centering vectors:
  0.000  0.000  0.000  0.000
  0.500  0.500  0.000  0.000

Symmetry operations compatible with the lattice and centering:
                              Symmetry operation             agreement factor
 m|s(0,1,0):      x1    -x2     x3    0.5+x4    4.528    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 2|0(0,1,0):     -x1     x2    -x3       -x4     8.206     XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
        -1:      -x1    -x2    -x3       -x4    12.611    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 m|0(0,1,0):      x1    -x2     x3        x4    47.818    XXXXXXXXXXXXXXXXXXXXXXXXX
 c|0(0,1,0):      x1    -x2 0.5+x3        x4    94.059    XXX
 c|s(0,1,0):      x1    -x2 0.5+x3    0.5+x4   95.673    XX

-------------------------------------------------
Space group derived from the symmetry operations:
-------------------------------------------------
 Centering vectors:
  0.000  0.000  0.000  0.000
  0.500  0.500  0.000  0.000

 Symmetry operations:
             -1:          -x1          -x2          -x3          -x4
              1:           x1           x2           x3           x4
     m|s(0,1,0):           x1          -x2           x3     0.500+x4
     2|0(0,1,0):          -x1           x2          -x3     0.500-x4

-----------------------------------------
Tentative space group symbol: C2/m(a0g)0s
-----------------------------------------
```

If the value of the keyword `searchsymmetry` is not `no`, the search for the symmetry operations is performed.

The agreement factor determines the degree of coincidence between the original and symmetry-transformed density. The expected value of the agreement factor for a random shift is 100%. The smaller the agreement factor, the better the match. First the agreement factors for individual generators used for the symmetry search are listed, and then the overall agreement factor that includes all symmetry operations of the space group.

```
############################################
# Search for the origin of the space group #
############################################

Agreement factors of individual generators:
Number  agreement
   4       0.68
   2       3.41

Overall agreement factor:   2.70
```

If a reference file is given (see keyword `referencefile`), info is written about the alignment of the current density and the reference density. The agreement factor of about 1% proves that the two densities are indeed very similar.

```
The density was aligned with the reference file, agreement  1.325%.
```

Final notes on the possibility to get more info with expanded log and record of the name of the input file and end of the calculation:

```
You can obtain more information about the reconstructed reflection phases by using 'expandedlog yes'.

Electron density written to file Cr2P2O7_sf.m81.


-------------------------------------------
End of the calculation: 16.JAN 2006, 16:47:15
-------------------------------------------
```

Chapter 6

Run-time interaction with the program

Although SUPERFLIP is primarily written as non-interactive, and all the instructions to the program can be given in the input file, it is occasionally useful to have the possibility to pass a command to the program during the execution. A limited set of commands is available for this purpose. To maintain the portability between platforms and independency on the peculiarities of different systems and compilers, the commands must be written to a file. The command file is checked by the program and if some known command is found, it is performed and the file is deleted. The standard name of the command file is `jobname.sfcom` (for definition of `jobname` see keyword `filebase`). The name of the command file can be changed by keyword `commandfile`. No multiple commands are allowed in the command file. The commands are case-sensitive. The most convenient way to pass a command to SUPERFLIP is to use command echo:

```
$ echo "command" > command-file_name
```

The allowed commands are:

- `density` *filename*: If this command is found, the current density is written to the file specified in the command and the iteration continues. The format of the file will correspond to the format of the (first) regular output density file.
- `stop`: If this command is found, the iteration is stopped, like if the maximal allowed number of cycles were exceeded. The output density and all output files are written.
- `perform cf|lde`: Allow a change of the method to low-density elimination or charge flipping.
- `delta` *value*: Sets the value of delta (see the corresponding keyword) to *value*.
- `weakratio` *value*: Sets the weakratio (see the corresponding keyword) to *value*.

Chapter 7

Handling of symmetry in Superflip

One of the main advantages of charge flipping is that a mistake in estimating the space group does not prevent the solution of the structure. This flexibility offers a range of possibilities to approach the problem of symmetry, and this chapter should give an overview of the possibilities offered by Superflip, and a guide to their proper and efficient use.

The general procedure is the following:

- The space group is read from the input file (keyword `symmetry - endsymmetry`)
- The reflections from the input file are averaged and expanded by the Laue group derived from the input space group. Systematically extinct reflections are allowed in the input. Thus, the preparation of the reflections list depends only on the Laue group, and not on the space group.

  It is always possible to average the reflections only in Laue group $\bar{1}$, and let Superflip derive the true space group later (see below), but averaging the data in the appropriate Laue group decreases noise and enhances data completeness, and therefore it is still preferable.
- The charge-flipping iteration works with the expanded list of reflections, and makes no use of the space-group information whatsoever. The result is a density, which is randomly shifted in the cell, and approximately obeys the crystal symmetry. At this moment, the structure is in principle solved, but the density has to be further processed to impose the symmetry and shift the origin appropriately.
- Depending on the values of the keyword `searchsymmetry` the density can be saved as it is after the calculation (`i.e.` randomly shifted and only with approximate symmetry), or the origin of the symmetry can be located, the density can be shifted to this origin, but not averaged, or, the default behavior, the origin of the space group is located, density is shifted, and the pixels that are equivalent in the space group are averaged. As a result, the density produced with the third option will ALWAYS obey the symmetry from the input file, even if it is not correct. How well is the symmetry present in the density can be judged from the so-called *symmetry agreement factors* that are printed out after the symmetry search. An example of the space group *Pcmn*:

```
Searching for the origin of the space group:

Agreement factors of individual generators:
Number  agreement
   2       0.78
   3       0.11
   4       0.47
Overall agreement factor:  0.76
```

The numbers of the generators relate to their position in the list of symmetry operations in the input file. Agreement factor is defined in analogy with the R-factor, *i.e.* the lower the better. Agreement factor 100 corresponds to a completely random density with no sign of the assumed symmetry. The precise values of the agreement factors depend very much on the quality of the data and on the particular structure, but in general values below 10 are a sign of a very good agreement, and values below 20 are still acceptable. The above example shows an excellent agreement, with agreement factors below 1.

If the user would input space group $Pmmm$ instead of the correct $Pcmn$, the calculation would still converge to the right solution, but the symmetry search would produce output similar to this:

```
Searching for the origin of the space group:

Agreement factors of individual generators:
Number  agreement
   2      61.89
   3       0.45
   4      72.80
Overall agreement factor: 61.84
```

This shows clearly, that the second and fourth symmetry operations from the list are not present in the density, but the third is. Obviously, the input symmetry is incorrect and it should be changed.

Sometimes the symmetry searching algorithm cannot find a common origin for all generators. This happens very often if the calculation does not converge, or if the symmetry is completely wrong. In such a case a warning occurs informing about the failure of the least-square procedure searching for the common origin:

```
Agreement factors of individual generators:
Number  agreement
  14      86.67
  15      82.62
  17     106.16
   4     100.45
Warning, a discrepancy in the least-squares solution is   6.41, normal
values are below 0.5.
The solution might be unreliable (no convergence or wrong symmetry...?)
Overall agreement factor:  97.90
```

7.1 Automatic derivation of symmetry from the density

Parallel to the general procedure described above SUPERFLIP can also derive the space group directly from the calculated scattering density using the keyword `derivesymmetry`. This procedure can be used as an independent check of the correctness of the assumed space

group, or as a way to determine the space group *ab initio*. To profit from this valuable option it is advisable to understand its principle, and use it with care and critical mind rather than blindly.

The procedure in deriving the symmetry is the following:

- First the lattice centering present in the density is detected.
- Then all possible symmetry operations compatible with the geometry of the lattice are identified. Only symmetry operations with rotational parts that bring a basic lattice vector onto itself or another basic lattice vector are taken into account.
- Then for each potential symmetry operation its position in the density and its agreement factor are calculated, and a list of symmetry operations with their symbols and agreement factors is printed in order of ascending agreement factor:

```
Checking the density for symmetry:

Symmetry operations compatible with the lattice and centering:
                     Symmetry operation             agreement factor
  m(0,1,0):              x1          -x2           x3    0.530    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  n(0,0,1):       0.500+x1     0.500+x2          -x3    1.822    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  c(1,0,0):           -x1           x2     0.500+x3    1.832    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
2_1(0,0,1):           -x1          -x2     0.500+x3    2.291    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
2_1(1,0,0):      0.500+x1          -x2          -x3    2.315    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
2_1(0,1,0):           -x1     0.500+x2          -x3    3.558    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
        -1:           -x1          -x2          -x3    4.034    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
  m(1,0,0):           -x1           x2           x3   62.667    XXXXXXXXXXXXXXXXXX
  2(0,0,1):           -x1          -x2           x3   62.972    XXXXXXXXXXXXXXXXXX
  m(0,0,1):            x1           x2          -x3   74.663    XXXXXXXXXXXXX
  2(1,0,0):            x1          -x2          -x3   74.877    XXXXXXXXXXXXX
  2(0,1,0):           -x1           x2          -x3   75.350    XXXXXXXXXXXX
  n(1,0,0):           -x1     0.500+x2     0.500+x3   92.841    XXXX
  b(1,0,0):           -x1     0.500+x2           x3   94.945    XXX
  c(0,1,0):            x1          -x2     0.500+x3   96.146    XX
  b(0,0,1):            x1     0.500+x2          -x3   96.311    XX
  n(0,1,0):       0.500+x1          -x2     0.500+x3  104.091    X
  a(0,1,0):       0.500+x1          -x2           x3  104.755    X
  a(0,0,1):       0.500+x1           x2          -x3  113.698    X
```

  The row of "X" at the end of each serves as a guide for eye to facilitate judging, where is the step between the relevant and irrelevant symmetry operations.

- Then a threshold is applied to the list, and only symmetry operations with agreement factor below the threshold value are further considered. The default value of the threshold is 25, but can be altered by the user (see description of the keyword `derivesymmetry`).
- Symmetry operations that passed the threshold are then checked, if they form a complete space group, and if they do not, missing symmetry operations are added. At the same time those symmetry operations are excluded that, if combined with other operation from the list with lower agreement factor, generate nontrivial lattice centering other than that detected previously as true centering vector. Such situation occurs very often in cases of pseudosymmetry. A completed list of symmetry operation is then written out in a format that can be copy-pasted in the input file for SUPERFLIP (without the symbols of the symmetry elements):

```
Space group derived from the symmetry operations:

 Symmetry operations:
              1:            x1           x2           x3
             -1:           -x1          -x2          -x3
       m(0,1,0):            x1    0.500-x2           x3
       n(0,0,1):      0.500+x1    0.500+x2    0.500-x3
       c(1,0,0):      0.500-x1           x2    0.500+x3
```

```
2_1(1,0,0):      0.500+x1          -x2      0.500-x3
2_1(0,0,1):      0.500-x1      0.500-x2      0.500+x3
2_1(0,1,0):           -x1      0.500+x2           -x3
```

- Finally, if the dimension of the density is 3, or if the structure is (3+1)D, `i.e.` the physical dimension is 3, and total dimension is 4 (see keywords `dimension` and `realdimension`), a tentative Herman-Mauguin symbol of the space group is derived. The word tentative means that SUPERFLIP derives the symbol algorithmically, and despite of big effort being put in making the result compliant with the conventions, the compliance cannot be guaranteed in all cases.

To conclude this section let me write a few words of warning against uncritical acceptance of the space group offered by SUPERFLIP. The derived space group depends on several factors, the main of them being the threshold for the agreement factor. It is impossible to determine the correct agreement factor for all situations automatically, and it is the responsibility of the user to critically review the result to check, if the applied threshold is correct. This is also the reason why the symmetry derivation produces relatively large amount of output, although just the space group symbol might seem sufficient in most cases. As an example, this is a listing obtained from one ortorhombic structure:

```
Symmetry operations compatible with the lattice and centering:
                          Symmetry operation              agreement factor
       2(0,0,1):           -x1          -x2           x3    2.850    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
       2(0,1,0):           -x1           x2          -x3    5.994    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
       2(1,0,0):            x1          -x2          -x3    6.079    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
       m(1,0,0):           -x1           x2           x3   12.755    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
       m(0,1,0):            x1          -x2           x3   12.843    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
       m(0,0,1):            x1           x2          -x3   15.986    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
            -1:           -x1          -x2          -x3   18.832    XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
     2_1(0,1,0):           -x1     0.500+x2          -x3   44.801    XXXXXXXXXXXXXXXXXXXXXXXXXXXX
       b(0,0,1):            x1     0.500+x2          -x3   50.982    XXXXXXXXXXXXXXXXXXXXXXXXXXXX
       b(1,0,0):           -x1     0.500+x2           x3   63.305    XXXXXXXXXXXXXXXXXX
     2_1(1,0,0):      0.500+x1          -x2          -x3   89.044    XXXXX
       a(0,1,0):      0.500+x1          -x2           x3   90.066    XXXXX
       n(0,0,1):      0.500+x1     0.500+x2          -x3   95.882    XX
     2_1(0,0,1):           -x1          -x2     0.500+x3   96.897    XX
       a(0,0,1):      0.500+x1           x2          -x3   97.089    X
       c(0,1,0):            x1          -x2     0.500+x3  111.953    X
       n(1,0,0):           -x1     0.500+x2     0.500+x3  113.036    X
       c(1,0,0):           -x1           x2     0.500+x3  115.549    X
       n(0,1,0):      0.500+x1          -x2     0.500+x3  126.091    X

Space group derived from the symmetry operations:

 Symmetry operations:
            1:            x1           x2           x3
           -1:           -x1          -x2          -x3
       m(1,0,0):          -x1           x2           x3
       m(0,1,0):           x1          -x2           x3
       m(0,0,1):           x1           x2          -x3
       2(0,0,1):          -x1          -x2           x3
       2(0,1,0):          -x1           x2          -x3
       2(1,0,0):           x1          -x2          -x3

***********************************************
Tentative space group symbol: Pmmm
***********************************************
```

At the first glance the symmetry seems to be clear, the agreement factors of the elements of the space group Pmmm are much lower than the rest of the elements. However, a second look shows that the two-fold axes have systematically better agreement factors than the mirror planes. And indeed, it turns out that the true symmetry of the structure is P222, but the heavy atoms (Pd and Sr) all obey the Pmmm symmetry, and only the light atoms (phosphorus in this case) break it.

Chapter 8

Superflip and powder data

Powder diffraction data represent a structure-solution challenge n its own. This chapter will introduce the concepts behind the powder option implemented in SUPERFLIP. For the detailed description of the input see the list of keywords. If the overlap of the reflections is low, it is possible to extract the intensities by LeBail or Pawley method and use SUPERFLIP in the normal "single crystal" mode. For such use no special modification to the single-crystal input file are necessary. On the other hand, if the overlap becomes large, the extracted intensities become very inaccurate and it is in general not possible to solve the structure directly from the extracted intensities. Then a method has to be found that will allow for an improvement of the intensities, or, as a special case, for repartitioning the integral intensity of a group of overlapping reflections among these reflections.

Up to now two methods have been devised for this purpose. One method (Wu, Leinenweber, Spence and O'Keeffe, 2006) uses the amplitudes of the "flipped" structure factors $G(\mathbf{h})$ (see Section 2) to repartition the intensities of the overlapping reflections. This method is not implemented in SUPERFLIP, because a few tests showed that it is useful only for very small structures and cannot compete with other structure solution methods. The other method has been suggested by Christian Baerlocher from ETH Zürich, and has been published in Baerlocher, McCusker and Palatinus (2007). If you are interested in the principles and background of the method, please refer to this publication. Here only the technical aspects of the implementation will be discussed.

The method is based on matching the histogram of the trial electron density to the expected histogram of the correct density. The underlying assumption is that the histogram does not critically depend on the details of the structure but only on the number and type of the atoms in the structure. Thus, the correct histogram can be estimated before the structure is actually solved. Given a density ρ, its histogram can be obtained by the following procedure:

The list of density values is sorted in ascending order, indexed from 0 to $N_{pix}-1$ (N_{pix} is the number of density pixels), and then the members of the list with index $(N_{pix}-1)i/N_c,\ i = 0,\ldots,N_c$ (N_c being the number of desired histogram classes) are selected, and their density values $\tau_i,\ i = 1,\ldots,N_c$ are stored. The density values τ_i represent the histogram of the density.

The expected histogram must be provided in the input file for Superflip (see keyword `histogram - endhistogram`). It is important that the number of classes is such that the contiguous values in the histogram are not too different - especially in the high-value region of the histogram. The expected histogram can be obtained either from a structure similar to the structure that is being solved, or from a randomly generated structure with the same composition as the unknown structure. Moreover, the Debye-Waller factor influences the histogram, and therefore a reasonable estimate of the Debye-Waller factor should be used when generating the structure for the histogram.

The algorithm is described in Figure 8.1. It starts as a normal charge-flipping iteration.

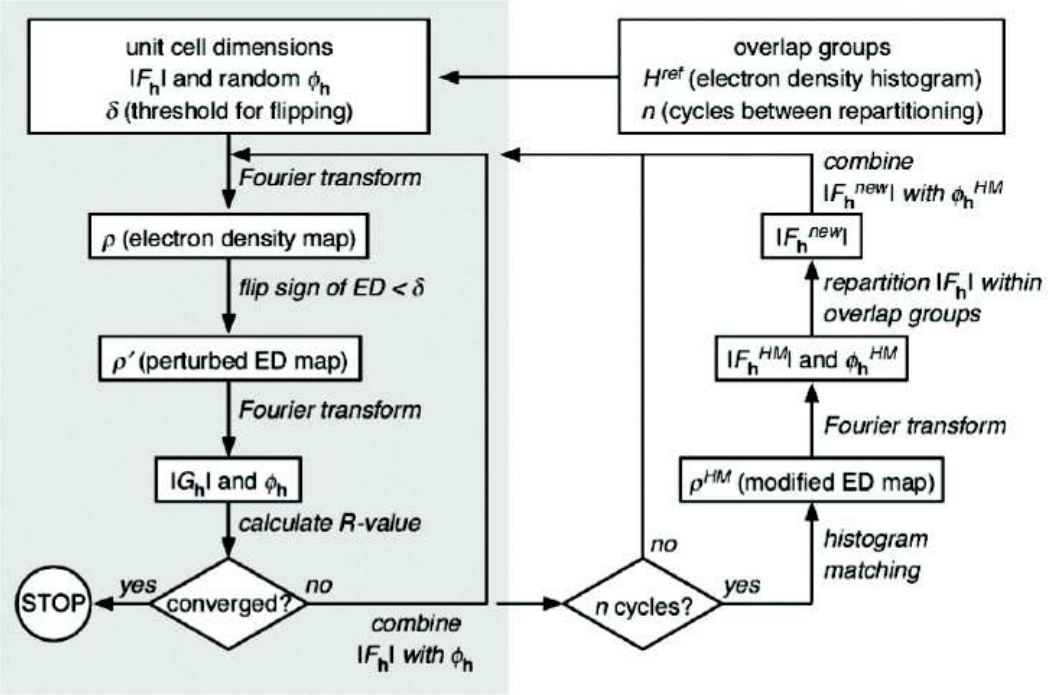


Figure 8.1: A flowchart of the charge flipping algorithm with histogram matching. The original charge flipping is shown shaded, the modifications are on the white background. Reproduced from Baerlocher et al. (2007).

When certain number of cycles is reached, the histogram-matching procedure is performed, and the intensities of the overlapping reflections are repartitioned according to the ratios of intensities obtained from the density modified by the histogram-matching procedure. Then the charge-flipping algorithm continues with the modified intensities.

Apart form the histogram there is another important information that must be present in the input file, and that is the indication, which reflections are overlapping. Currently there are two ways to do that. The overlapping reflections can be given the same group number. An external program must be used to decide, which reflections are overlapping, and assign the group numbers. The the other option is to supply the FWHM of each reflection and let Superflip calculate the overlap groups. See the keyword `dataformat` for details on how the group numbers or FWHM must be defined. To calculate the overlap groups from FWHM Superflip needs also information about the wavelength and the overlap threshold (see the keywords `lambda` and `fwhmseparation`).

Finally: please note that the charge flipping algorithm with powder data cannot really compete with the direct-space based methods like FOX, especially not if the resolution of the powder diagram is low, provided the contents of the structure is known. On the other hand it is quite successful in true *ab initio* solution of structures with good resolution, especially those measured with synchrotron radiation.

Chapter 9

Some know-how or what to do if things go wrong

This chapter is intended to help you start using SUPERFLIP, getting your structures as quickly as possible and understanding potential problems. I will try to summarize my experience with charge flipping here. So, please, be aware that the following observations are necessarily incomplete, maybe even inaccurate or partially wrong, because they apply to a limited, albeit large number of structures solved by myself. Having said that, I still believe these notes will be helpful to everybody who is just getting started with charge flipping.

9.1 The value of δ

δ is the crucial parameter of the whole calculation. Unfortunately, no way has been found up to now to determine δ for a given data set a priori. δ depends on the scale of the amplitudes, on the resolution, on the value of the Debye-Waller factor and, of course, on the contents of the unit cell. Despite of that there are ways to determine δ empirically. The most reliable way that I have discovered is to compare the amount of total charge and flipped charge (see subsection 5.2.4 for definition). In a typical iteration the initial three transition cycles are followed by a plateau of several to several thousand cycles before the convergence sets on. On this plateau, many characteristics of the iteration remain approximately constant, including the R-value, the total charge and the flipped charge. I have made the observation that the value total/flipped charge should be in an ideal case somewhat less that one. If the value is too large (too little flipped charge), δ must be increased and *vice versa*. This procedure can be easily automated and the automatic determination of δ works just in this way.

This criterion is closely related to the fact that the realistic structures have approximately the same amount of "signal/background" ratio, i.e. the proportion of pixels bearing the significant density tends to be about the same in realistic structures. Of course, this is not true exactly, the organic structures will have more significant pixels that a structure with heavy atoms and the same unit-cell size. This can be taken into account and δ can be correspondingly modified, if the automatic search does not lead to convergence. The larger is the proportion of the significant pixels, the larger must be the ratio of the total/flipped charge.

Occasionally you will encounter a rapid convergence, which however does not yield any sensible map. This can be easily recognized on a very small R-value (usually around 1% or even less). In addition to that either the amount of the total charge or the amount of the flipped charge is close to zero. In the first case the δ is way too large, in the second case it is way too small. This can be easily understood. If δ is too large, virtually all pixels in the starting density will be flipped, which yields almost $\rho_{new} = -\rho_{old}$, so almost

no perturbation is performed, the amplitudes remain almost the same and the R-value is very low despite of the density being completely random. Similarly, if δ is too small, only very little perturbation is performed, the amplitudes also do not change significantly and the R-value again stays low. The program does not detect the false convergence, so if you use a fixed value of δ, be sure to check if the resulting R-value and the total and flipped charge have reasonable values.

9.2 The symmetry

As already mentioned, charge flipping reconstructs the density without any other information about the symmetry than what is contained in the amplitudes of the structure factors. Thus, the symmetry is contained in the resulting density, but only approximately and the origin is shifted. SUPERFLIP therefore searches for the origin of the space group. This usually works pretty well. Problems occur only if the disturbance of the symmetry is too small to exhibit itself in the approximate density. Therefore, do not completely rely on the symmetry search in cases of superstructures with small deviation of the superstructure from the basic structure. In case of doubts, whether the symmetry operations are correctly located, rerun SUPERFLIP and do not use `searchsymmetry average`, but `searchsymmetry shift`. This option will try to locate the origin, but it will not average the density, but instead of that it will only shift it. So the original density remains preserved and you can analyse it for small symmetry disturbances.

Another problem can occur, if you work with a structure that is almost centrosymmetric, but a substructure breaks the centrosymmetry. In that case it is likely that charge flipping will find a centrosymmetric structure that will contain both orientations of the non-centrosymmetric substructure superimposed. On the other hand, a robust acentricity of the structure is properly reconstructed by charge flipping.

9.3 The convergence

It is usually very easy to assess the convergence by visually inspecting the curve of the R-value or peakiness vs. number of iteration steps. The convergence is marked by a quick drop of the R-value from the plateau down to values typically between 20 and 30%. It is somewhat more difficult to automate the recognition procedure. SUPERFLIP can recognize the convergence in most of the cases, but it fails occasionally. The two cases most usual reasons for the failure of the detection algorithm are:

- The convergence is too fast and no plateau has developed that could be used as a reference.
- The step between the converged and non-converged values of the indicators is too small to be recognized.

If you encounter this situation, just run SUPERFLIP again with a limited number of cycles (see section 3.3) and select the number of cycles so that the convergence is reached within this number.

9.4 Charge flipping converges, but I cannot refine the structure!

Before trying to pinpoint the problem, one has to keep in mind one thing: For most of the easy to moderately difficult structures the direct methods are the method of first choice. Therefore, charge flipping is tried most frequently on structures that direct methods fail on. Such a selection of structures is likely to have some intrinsic problem, often with the data

quality or with twinning. If charge flipping converges, but the resulting structure seems to make no sense or cannot be refined well, it is an indication of one of the following issues:

- The data correspond to a twinned structure. This is actually the most probable explanation.
- The data are only a subset of all reflections, some superstructure reflections have been omitted and thus the result is a superposition structure.
- The structure has a non-centrosymmetric substructure and charge flipping reconstructed a superposition structure with both non-centrosymmetric substructures superimposed.

With the exception of the last item, which is quite rare, the effect of "convergence without refinement" is indicative of a problem in the data set rather that with the structure being too difficult to solve.

Bibliography

Baerlocher, C., McCusker, L. and Palatinus, L. (2007), 'Charge flipping combined with histogram matching to solve complex crystal structures from powder diffraction data', *Z. Kristalogr.* **222**, 47–53.

CCP4 (1994), 'Collaborative computationla project number 4 (ccp4)', *Acta Cryst. D* **50**, 760–764.

Oszlányi, G. and Sütő, A. (2004), '*Ab initio* structure solution by charge flipping', *Acta Cryst. A* **60**, 134–141.

Oszlányi, G. and Sütő, A. (2005), '*Ab intitio* structure solution by charge flipping. ii. use of weak reflections', *Acta Cryst. A* **61**, 147–152.

Palatinus, L. (2004), '*Ab intitio* determination of incommensurately modulated structures by charge flipping in superspace', *Acta Cryst. A* **60**, 604–610.

Petříček, V., Dušek, M. and Palatinus, L. (2000), *The crystallographic computing system JANA2000*, Institute of Physics, Praha, Czech Republic.

Shiono, M. and Woolfson, M. M. (1992), 'Direct-space methods in phase extension and phase determination. i. low-density elimination.', *Acta Cryst. A* **48**, 451–456.

Wu, J. S., Leinenweber, K., Spence, J. C. H. and O'Keeffe, M. (2006), 'Ab initio phasing of x-ray powder pattern by charge flipping', *Nature Materials* **5(8)**, 647–652.

Wu, J. S., Spence, J. C. H., O'Keeffe, M. and Groy, T. L. (2004), *Acta Cryst. A* **60**, 326–330.

Zhang, K. Y. J. and Main, P. (1990), 'Histogram matching as a new density modification techique for phase refinement and extension of protein molecules', *Acta Cryst. A* **46**, 41–46.