cctbx tools for CIF & parameter optimization

Richard J. Gildea

DIALS3:
Detectors, Data Collection, and Software

February 11th 2012

BERKELEY LAB
Lawrence Berkeley
National Laboratory

What is CIF?

- Crystallographic Information File (Framework)
- Based upon Self-defining Text Archive and Retrieval (STAR) file format (Hall, 1991; Hall & Spadaccini, 1994)
- Adopted by IUCr as preferred format for data exchange in 1990
- Flexible and extensible file structure
- Any type of text or numerical data can be arranged in any order
- Dictionary-based validation

CIF applications

- Submission of small molecule (i.e. non-macromolecules) crystal structures to Acta Crystallographica journals require coordinates and reflection data in CIF format
  - Automated validation of small molecule crystal structures using checkCIF
- CBF format (extension of CIF) for storing binary images
- Various uses of mmCIF format in macromolecular crystallography
  - Chemical components dictionary describes all residues and small molecule ligands from PDB in mmCIF format
  - CCP4 monomer library provides restraints for refinement in mmCIF format
  - Reflection data from PDB only available in mmCIF format
  - Future: PDB to require (prefer) submission of model coordinates and reflection data in mmCIF format

Adoption of PDBx/mmCIF format

- Sept 2011: formation of a working group for the adoption of the PDBx/mmCIF format for deposition and as an exchange format between programs for macromolecular crystallography
- Old-style PDB format increasingly problematic for large structures and for structures with complex chemistry
- Switch to PDBx/mmCIF format, particularly for deposition
- Facilitate capture of more information in deposition and archiving
- Enable direct use of PDBx/mmCIF format files in major macromolecular crystallographic software tools
- Involvement from developers of Phenix, CCP4, REFMAC, COOT, BUSTER, wwPDB, CCPN
- Agreement of developers to support direct input and output of mmCIF format by Jan 2013.

Relevance of CIF to DIALS

- CBF/ImgCIF
  - Prototype pyCBFImage detector class using iotbx.cif
- Exporting/archiving of unmerged reflections
  - iotbx.cif provides tools for exporting and extracting miller arrays to/from CIF objects
- Reporting of metadata from data processing for archiving with PDB deposition
  - Who wants to write yet another log file parser?

Formal CIF grammar

syntactic unit	syntax	case sensitive?
Basic Structure of a CIF		
<CIF>	<Comments>? <WhiteSpace>? { <DataBlock> { <WhiteSpace> <DataBlock> }* { <WhiteSpace> }? }?	yes
<DataBlock>	<DataBlockHeading> {<WhiteSpace> { <DataItems> | <SaveFrame>} }*	yes
<DataBlockHeading>	<DATA_> { <NonBlankChar> }+	no
<SaveFrame>	<SaveFrameHeading> { <WhiteSpace> <DataItems> }+ <WhiteSpace> <SAVE_>	yes
<SaveFrameHeading>	<SAVE_> { <NonBlankChar> }+	no
<DataItems>	<Tag> <WhiteSpace> <Value> | <LoopHeader> <LoopBody>	yes
<LoopHeader>	<LOOP_> {<WhiteSpace> <Tag>}+	no
<LoopBody>	<Value> { <WhiteSpace> <Value> }*	yes
Reserved Words		
<DATA_>	{'D' | 'd'} {'A' | 'a'} {'T' | 't'} {'A' | 'a'} '_'	no
<LOOP_>	{'L' | 'l'} {'O' | 'o'} {'O' | 'o'} {'P' | 'p'} '_'	no
<GLOBAL_>	{'G' | 'g'} {'L' | 'l'} {'O' | 'o'} {'B' | 'b'} {'A' | 'a'} {'L' | 'l'} '_'	no
<SAVE_>	{'S' | 's'} {'A' | 'a'} {'V' | 'v'} {'E' | 'e'} '_'	no
<STOP_>	{'S' | 's'} {'T' | 't'} {'O' | 'o'} {'P' | 'p'}'_'	no
Tags and Values		
<Tag>	'_'{ <NonBlankChar>}+	no
<Value>	{ '.' | '?' | <Numeric> | <CharString> | <TextField> }	yes
Numeric Values		
<Numeric>	{ <Number> | <Number> '(' <UnsignedInteger> ')' }	no
<Number>	{<Integer> | <Float> }	no
<Integer>	{ '+' | '-' }? <UnsignedInteger>	no
<Float>	{ <Integer><Exponent> | { {'+'|'-'} ? { {<Digit>} * '.' <UnsignedInteger> } | { <Digit>} + '.' } } {<Exponent>} ? } }	no
<Exponent>	{ {'e' | 'E' } | {'e' | 'E' } { '+' | '- ' } } <UnsignedInteger>	no
<UnsignedInteger>	{ <Digit> }+	no
<Digit>	{ '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9' }	no
Character Strings and Text Fields		
<CharString>	<UnquotedString> | <SingleQuotedString> | <DoubleQuotedString>	yes
<eol><UnquotedString>	<eol><OrdinaryChar> {<NonBlankChar>}*	yes
<noteol><UnquotedString>	<noteol>{<OrdinaryChar>|';'} {<NonBlankChar>}*	yes
<SingleQuotedString> <WhiteSpace>	<single_quote>{<AnyPrintChar>}* <single_quote> <WhiteSpace>	yes
<DoubleQuotedString> <WhiteSpace>	<double_quote> {<AnyPrintChar>}* <double_quote> <WhiteSpace>	yes
<TextField>	{ <SemiColonTextField> }	yes
<eol><SemiColonTextField>	<eol>';' { {<AnyPrintChar>}* <eol> {{<TextLeadChar> {<AnyPrintChar>}*}? <eol>}* } ';'	yes

http://www.iucr.org/resources/cif/spec/version1.1/cifsyntax#bnf

CIF format

- **data name** (tag)
  - string of characters beginning with an underscore character (no whitespace)
  - identifier of the content of an associated data value.
- **data value**
  - string of characters representing a particular item of information
  - single numerical value; a letter, word or phrase; extended text; any coherent unit of data (e.g. an image).
- **data item**
  - specific piece of information defined by a data name and its associated data value(s)
  - Data names serve to index data values and are case-insensitive.
- whitespace (unquoted) serves as a delimiter between tokens

CIF format

- **data block**
  - highest-level component of a CIF, contains data items
  - Identified by data-block header, starting with characters `data_`, e.g. `data_foo`
- **looped list**
  - Vector (list) or matrix (table) of set of data items
  - Keyword `loop_` followed by one or more data names and one or more data values (must be an exact multiple of number of data names)
  - Data values in row-major order
  - May not be nested

Unit cell and symmetry section

data block header →

tag-value pairs →

loop_ keyword →

data names →

data values →

```
data_3adr

_cell.length_a                          163.97
_cell.length_b                          45.23
_cell.length_c                          110.89
_cell.angle_alpha                       90.0
_cell.angle_beta                        131.64
_cell.angle_gamma                       90.0
_cell.volume                            614608.19

loop_
  _space_group_symop.id
  _space_group_symop.operation_xyz
   1 x,y,z
   2 -x,y,-z
   3 x+1/2,y+1/2,z
   4 -x+1/2,y+1/2,-z

_space_group.crystal_system             monoclinic
_space_group.IT_number                  5
_space_group.name_H-M_alt               'C 1 2 1'
_space_group.name_Hall                  ' C 2y'
```

Atomic coordinates

```
ATOM      1  N   ASP L   1A     11.311  19.158  20.128  1.00 30.02           N
ATOM      2  CA  ASP L   1A     10.104  18.681  19.448  1.00 29.12           C
ATOM      3  C   ASP L   1A      9.872  17.159  19.556  1.00 26.73           C
ATOM      4  O   ASP L   1A      8.811  16.652  19.172  1.00 21.71           O
ATOM      5  CB  ASP L   1A     10.144  19.086  17.973  1.00 38.70           C
ATOM      6  CG  ASP L   1A      8.760  19.339  17.395  1.00 44.90           C
ATOM      7  OD1 ASP L   1A      7.777  19.324  18.169  1.00 44.51           O
ATOM      8  OD2 ASP L   1A      8.661  19.578  16.171  1.00 50.05           O
```

PDB

```
loop_
  _atom_site.group_PDB
  _atom_site.id
  _atom_site.label_atom_id
  _atom_site.label_alt_id
  _atom_site.label_comp_id
  _atom_site.auth_asym_id
  _atom_site.auth_seq_id
  _atom_site.pdbx_PDB_ins_code
  _atom_site.Cartn_x
  _atom_site.Cartn_y
  _atom_site.Cartn_z
  _atom_site.occupancy
  _atom_site.B_iso_or_equiv
  _atom_site.type_symbol
  _atom_site.pdbx_formal_charge
  _atom_site.label_asym_id
  _atom_site.label_entity_id
  _atom_site.label_seq_id
  _atom_site.pdbx_PDB_model_num
  ATOM       1  N     .  ASP  L    1  A  11.31147   19.15776  20.12829  1.000  30.02111  N   ?  A  ?    1  1
  ATOM       2  CA    .  ASP  L    1  A  10.10373   18.68105  19.44836  1.000  29.12464  C   ?  A  ?    1  1
  ATOM       3  C     .  ASP  L    1  A   9.87214   17.15878  19.55617  1.000  26.73154  C   ?  A  ?    1  1
  ATOM       4  O     .  ASP  L    1  A   8.81111   16.65235  19.17214  1.000  21.71081  O   ?  A  ?    1  1
  ATOM       5  CB    .  ASP  L    1  A  10.14403   19.08618  17.97261  1.000  38.70228  C   ?  A  ?    1  1
  ATOM       6  CG    .  ASP  L    1  A   8.75963   19.33936  17.39469  1.000  44.90010  C   ?  A  ?    1  1
  ATOM       7  OD1   .  ASP  L    1  A   7.77712   19.32440  18.16915  1.000  44.50718  O   ?  A  ?    1  1
  ATOM       8  OD2   .  ASP  L    1  A   8.66147   19.57786  16.17106  1.000  50.05253  O   ?  A  ?    1  1
```

mmCIF

REMARK 200 – data collection

```
REMARK 200 EXPERIMENTAL DETAILS
REMARK 200  EXPERIMENT TYPE                : X-RAY DIFFRACTION
REMARK 200  DATE OF DATA COLLECTION        : NULL
REMARK 200  TEMPERATURE           (KELVIN) : 100
REMARK 200  PH                             : 8.5
REMARK 200  NUMBER OF CRYSTALS USED        : 1
REMARK 200
REMARK 200  SYNCHROTRON              (Y/N) : Y
REMARK 200  RADIATION SOURCE               : ALS
REMARK 200  BEAMLINE                       : 8.3.1
REMARK 200  X-RAY GENERATOR MODEL          : NULL
REMARK 200  MONOCHROMATIC OR LAUE    (M/L) : M
REMARK 200  WAVELENGTH OR RANGE        (A) : 1.8927
REMARK 200  MONOCHROMATOR                  : Y
REMARK 200  OPTICS                         : NULL
REMARK 200
REMARK 200  DETECTOR TYPE                  : CCD
REMARK 200  DETECTOR MANUFACTURER          : ADSC QUANTUM 210R
REMARK 200  INTENSITY-INTEGRATION SOFTWARE : MOSFLM
REMARK 200  DATA SCALING SOFTWARE          : SCALA
REMARK 200
REMARK 200  NUMBER OF UNIQUE REFLECTIONS   : 6052
REMARK 200  RESOLUTION RANGE HIGH      (A) : 2.900
REMARK 200  RESOLUTION RANGE LOW       (A) : 20.000
REMARK 200  REJECTION CRITERIA  (SIGMA(I)) : 0.000
REMARK 200
REMARK 200 OVERALL.
REMARK 200  COMPLETENESS FOR RANGE     (%) : 99.7
REMARK 200  DATA REDUNDANCY                : 3.700
REMARK 200  R MERGE                    (I) : 0.08700
REMARK 200  R SYM                      (I) : NULL
REMARK 200  <I/SIGMA(I)> FOR THE DATA SET  : 11.1000
REMARK 200
REMARK 200 IN THE HIGHEST RESOLUTION SHELL.
REMARK 200  HIGHEST RESOLUTION SHELL, RANGE HIGH (A) : 2.90
REMARK 200  HIGHEST RESOLUTION SHELL, RANGE LOW  (A) : 3.06
REMARK 200  COMPLETENESS FOR SHELL     (%) : 99.9
REMARK 200  DATA REDUNDANCY IN SHELL       : 3.60
REMARK 200  R MERGE FOR SHELL          (I) : 0.40600
REMARK 200  R SYM FOR SHELL            (I) : NULL
REMARK 200  <I/SIGMA(I)> FOR SHELL         : 3.100
REMARK 200
REMARK 200 DIFFRACTION PROTOCOL: SINGLE WAVELENGTH
REMARK 200 METHOD USED TO DETERMINE THE STRUCTURE: MOLECULAR REPLACEMENT
REMARK 200 SOFTWARE USED: MOLREP
REMARK 200 STARTING MODEL: PDB ENTRY 1MRU
REMARK 200
REMARK 200 REMARK: NULL
```

```
_diffrn.id                     1
_diffrn.ambient_temp           100
_diffrn.ambient_temp_details   ?
_diffrn.crystal_id             1
#
_diffrn_detector.diffrn_id              1
_diffrn_detector.detector               CCD
_diffrn_detector.type                   'ADSC QUANTUM 210r'
_diffrn_detector.pdbx_collection_date   ?
_diffrn_detector.details                ?
#
_diffrn_radiation.diffrn_id                         1
_diffrn_radiation.wavelength_id                     1
_diffrn_radiation.pdbx_monochromatic_or_laue_m_l    M
_diffrn_radiation.monochromator                     Y
_diffrn_radiation.pdbx_diffrn_protocol              'SINGLE WAVELENGTH'
_diffrn_radiation.pdbx_scattering_type              x-ray
#
_diffrn_radiation_wavelength.id           1
_diffrn_radiation_wavelength.wavelength   1.8927
_diffrn_radiation_wavelength.wt           1.0
#
_diffrn_source.diffrn_id                    1
_diffrn_source.source                       SYNCHROTRON
_diffrn_source.type                         'ALS BEAMLINE 8.3.1'
_diffrn_source.pdbx_synchrotron_site        ALS
_diffrn_source.pdbx_synchrotron_beamline    8.3.1
_diffrn_source.pdbx_wavelength              1.8927
_diffrn_source.pdbx_wavelength_list         1.8927
#
_reflns.number_obs                   6052
_reflns.percent_possible_obs         99.7
_reflns.pdbx_Rmerge_I_obs            0.08700
_reflns.pdbx_Rsym_value              ?
_reflns.pdbx_netI_over_sigmaI        11.1000
_reflns.pdbx_redundancy              3.7
#
_reflns_shell.d_res_high             2.90
_reflns_shell.d_res_low              3.06
_reflns_shell.percent_possible_all   99.9
_reflns_shell.Rmerge_I_obs           0.40600
_reflns_shell.meanI_over_sigI_obs    3.1000
_reflns_shell.pdbx_redundancy        3.6
```

iotbx.cif

- iotbx.cif module provides fast, generic parsing and handling of **all types** of CIF files
- Uses ANTLR parser generator to handle lexing/parsing (C/C++)
- Python dictionary-like interface to internal CIF object
- Provides comprehensive tools for input, output and validation of CIFs
- Interconversion between CIF and cctbx crystallographic objects (xray.structure, miller.array, pdb.hierarchy)
- Native support for mmCIF model and reflection files in most Phenix programs

iotbx.cif: a comprehensive CIF toolbox. *J. Appl. Cryst.* (2011). 44, 1259-1263.

iotbx.cif: input

```
import iotbx.cif
from cctbx.array_family import flex

file_name = "1akg.cif”
cif_object = iotbx.cif.reader(file_path=file_name).model()

cif_block = cif_object["1AKG"]
space_group_name = cif_block["_symmetry.space_group_name_H-M"]
print space_group_name

>>> P 21 21 21

r_free = cif_block["_refine.ls_R_factor_R_free"]
r_work = cif_block["_refine.ls_R_factor_R_work"]
print r_work, r_free

>>> 0.147 0.157

atom_names = cif_block["_atom_site.label_atom_id"]
resnames = cif_block["_atom_site.label_comp_id"]
cart_x = flex.double(cif_block["_atom_site.Cartn_x"])
cart_y = flex.double(cif_block["_atom_site.Cartn_y"])
cart_z = flex.double(cif_block["_atom_site.Cartn_z"])

for i in range(len(atom_names)):
  print "%4s %3s %6.3f %6.3f %6.3f" %(
    atom_names[i], resnames[i], cart_x[i], cart_y[i], cart_z[i])

>>>
   N GLY  0.504 -0.494  0.924
  CA GLY  1.272  0.589  0.277
   C GLY  1.700  1.614  1.301
   O GLY  1.434  1.460  2.496
...
```

iotbx.cif: extract useful objects

```
import iotbx.cif
from cctbx.array_family import flex

# this will work for small molecule structure files
# returns a dictionary of xray.structure objects, one per data block,
# dictionary key is data block name
xray_structures = iotbx.cif.reader(
  file_path=file_path).build_crystal_structures()

# this will work for small molecule and mmcif-format reflections files
# returns a list of all miller arrays found in the file
miller_arrays = iotbx.cif.reader(file_path=file_path).as_miller_arrays()
for array in miller_arrays:
  array.show_comprehensive_summary()

# extract pdb.input and pdb.hierarchy objects
import iotbx.pdb.mmcif
pdb_input = iotbx.pdb.mmcif.cif_input(file_name=file_path)
pdb_hierarchy = pdb_input.construct_hierarchy()
```

iotbx.cif: output

```
from iotbx.cif import model

cif_object = model.cif()
cif_block = model.block()
cif_object["3adr"] = cif_block

# given cctbx space_group objects:

space_group_type = quartz_structure.space_group_info().type()
cif_block["_space_group.crystal_system"] = space_group.crystal_system().lower()
cif_block["_space_group.IT_number"] = space_group_type.number()
cif_block["_space_group.name_H-M_alt"] = space_group_type.lookup_symbol()
cif_block["_space_group.name_Hall"] = space_group_type.hall_symbol()

symop_loop = model.loop(header=("_space_group_symop.id",
                                "_space_group_symop.operation_xyz"))
for symop_id, symop in enumerate(space_group):
  symop_loop.add_row((symop_id + 1, symop.as_xyz()))
cif_block.add_loop(symop_loop)

print cif_object

>>> data_3adr
>>> loop_
>>>   _space_group_symop.id
>>>   _space_group_symop.operation_xyz
>>>    1 x,y,z
>>>    2 -x,y,-z
>>>    3 x+1/2,y+1/2,z
>>>    4 -x+1/2,y+1/2,-z

>>> _space_group.crystal_system       monoclinic
>>> _space_group.IT_number            5
>>> _space_group.name_H-M_alt         'C 1 2 1'
>>> _space_group.name_Hall            ' C 2y'
```

CIF summary

- Extremely flexible format that can be used to present many different kinds of data
- PDBx/mmCIF format to become increasingly important
  - All major refinement software to read/write mmCIF coordinate and reflection files by Jan 2013?
  - PDB moving towards mmCIF for deposition (no target date?)
  - Latest version of Phenix can read and write mmCIF coordinate and reflection files
- iotbx.cif provides generic tools for handling of CIF files, as well as specialised tools for handling reflection files, structure files and much more

Numerical optimisation: the short version

Gradient-free minimisers

- (semi-)global optimisers (Peter Zwart)
  - scitbx.simplex
  - scitbx.direct_search_simulated_annealing
  - Cross-entropy methods
  - Covariance matrix adaptation – evolution strategy
  - Differential evolution
- Improved radius of convergence (especially for non-convex functions) at cost of slower rates of convergence

Liu, Poon, Chen, Grosse-Kunstleve & Zwart, Computational Crystallography Newsletter (2012). 3, 22–27.

Derivative-based minimisers

- L-BFGS – scitbx.lbfgs
  - Quasi-Newton method
  - Uses first derivatives only, builds up estimate of inverse Hessian using the BFGS update formula
  - Fundamental to phenix.refine
- Gauss-Newton method – scitbx.lstbx
  - Exploits special structure of nonlinear least squares problem to approximate matrix of second derivatives (Hessian) using first derivatives of the residuals (the Jacobian matrix)

Derivative-based minimisers

- Newton-type minimisers, requiring explicit calculation of n x n matrix of second derivatives (Hessian)
- Damped Newton's method
  - scitbx.minimizers.damped_newton
- Newton with Moré and Thuente line search
  - scitbx.minimizers.newton_more_thuente_1994

Acknowledgements

- LBNL/CCI
  - Paul Adams
  - Pavel Afonine
  - Aaron Brewster
  - Nat Echols
  - Ralf Grosse-Kunstleve
  - Jeff Headd
  - Johan Hattne
  - Nigel Moriarty
  - Nick Sauter
- Durham University
  - Judith Howard
  - Luc Bourhis
  - Oleg Dolomanov
  - Horst Puschmann

BERKELEY LAB Lawrence Berkeley National Laboratory

Phenix

Numerical optimisation: the longer version

Numerical optimisation

- Given the smooth function *f(x)* find the unconstrained minimum

$$\min_x f(x)$$

- What is a solution?
  - A point x^* is a *global minimiser* if

$$f(x^*) \leq f(x) \text{ for all } x$$

  - A point x^* is a *local minimiser* if there is a neighbourhood N of x^* such that

$$f(x^*) \leq f(x) \text{ for all } x \in \mathrm{N}$$

Numerical optimisation

- Taylor series: second order approximation

$$f(x+p) \approx f(x) + \nabla f(x)^T p + \tfrac{1}{2} p^T \nabla^2 f(x) p$$

- If x^* is a local minimiser then $\nabla f(x^*) = 0$

  $\nabla^2 f(x)$ is positive semidefinite

- Matrix B is positive semidefinite if

$$p^T B p \geq 0 \text{ for all p}$$

Optimisation algorithms

- In order to find the values of the parameters that minimise the function f(x), use some algorithm that generates a series of search directions, p_k, and step lengths, α_k, which updates the parameters according to

$$x_{k+1} = x_k + \alpha_k p_k$$

Optimisation algorithms

- *Line search*: choose a direction p_k and perform one-dimensional search along this direction to decide a step length α

$$\min_{\alpha>0} f(x_k + \alpha p)$$

- *Trust region*: construct model function m_k, and search for minimiser of m_k in some region around x_k

$$\min_{p} m_k(x_k + p)$$

Choosing a search direction

- Search direction often determined according to equation of the form

$$p_k = -B_k^{-1} \nabla f_k$$

- The simplest case, $B_k = I$, the identity matrix results in the method of *steepest descent*

$$p_k = -\nabla f_k \,/\, \left\| -\nabla f_k \right\|$$

  - Large radius of convergence
  - Can have slow rate of convergence

Choosing a search direction: Newton's Method

- Newton direction: B_k is the exact Hessian matrix of second derivatives

$$p_k = -\nabla^2 f_k^{-1} \nabla f_k$$

- Reliable when difference between true function *f(x)* and quadratic model *m(p)* is not too large
- Near minimum all second derivatives must be positive
- Near maximum second derivatives are negative
- Curvatures become zero at inflection points that surround each local minimum
- If Hessian is not positive definite then its inverse may not exist and Newton direction undefined

Choosing a search direction: Newton's Method

- Explicit calculation of n x n matrix of second derivatives
- Doesn't necessarily require inversion of n x n matrix – e.g. Cholesky decomposition for solution of a system of linear equations, but still computationally expensive – $O(n^3)$
- Only suitable when the model is already close to the minimum
- Not feasible for except for small molecules and high-quality structures of small proteins

Choosing a search direction: quasi-Newton methods

- Use approximation B_k in place of the true Hessian

$$p_k = -B_k^{-1}\nabla f_k$$

- Approximation B_k updated after each iteration
- Do not require computation of Hessian
- Superlinear rate of convergence
- Changes in the gradient provide information about the second derivative of f along the search direction

Choosing a search direction: quasi-Newton methods

- B_k chosen to satisfy the *secant equation* $B_{k+1}s_k = y_k$

  where $s_k = x_{k+1} - x_k, \quad y_k = \nabla f_{k+1} - \nabla f_k$

- *BFGS* update formula:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$$

  - Hessian approximation guaranteed to be positive definite
  - Alternatively update the inverse Hessian, $H_k = B_k^{-1}$, directly to avoid need for matrix inversion
  - Requires initial approximation inverse Hessian, H_o

Limited-memory BFGS

- *BFGS* method requires storage of dense n x n matrix
- Each step of BFGS method has the form

$$x_{k+1} = x_k - \alpha_k H_k \nabla f_k$$

- L-BFGS algorithm updates a modified version of H_k implicitly by storing m vector pairs $\{s_i, y_i\}$, and obtain the product $H_k \nabla f_k$ by a series of inner products and vector summations

Limited-memory BFGS

- L-BFGS algorithm computes the product $H_k \nabla f_k$ using a two-loop recursion that requires $4mn$ multiplications and n additions
- Storage requirements are $2mn + 4n$
- Provide initial inverse Hessian, H_o
- Common choice is some multiple $\gamma_k I$ of the identity matrix

$$\gamma_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}$$

Limited-memory BFGS

- H_o can be chosen to be different at each step
- If H_o is some multiple of the identity matrix, then the first step taken is in the steepest descent direction
- Consequently scaling of variables is important

Nonlinear Least-Squares

- Minimise function of the form

$$f(x) = \frac{1}{2}\sum_{j=1}^{m} r_j^2(x)$$

- Gradients can be written in terms of the Jacobian matrix,

$$J(x) = \left[\frac{\partial r_j}{\partial x_i}\right]_{\substack{j=1,\ldots,m \\ i=1,\ldots,n}}$$

$$\frac{\partial f}{\partial x_i} = \sum_{j=1}^{m} r_j(x)\frac{\partial r_j}{\partial x_i}(x) \qquad \nabla f = J(x)^T f(x)$$

Nonlinear Least-Squares

- The second derivatives are calculated by

$$\frac{\partial^2 f}{\partial x_j \partial x_k} = \sum_{i=1}^{m} \left[\frac{\partial r_i}{\partial x_j}(x) \frac{\partial r_i}{\partial x_k}(x) + r_i(x) \frac{\partial^2 r_i}{\partial x_j \partial x_k}(x) \right]$$

- In the limit of small residuals this becomes

$$\frac{\partial^2 f}{\partial x_j \partial x_k} = \sum_{i=1}^{m} \frac{\partial r_i}{\partial x_j}(x) \frac{\partial r_i}{\partial x_k}(x)$$

$$\nabla^2 f(x) = J(x)^T J(x)$$

Nonlinear Least-Squares

- Search direction is found by solving

$$p_k = -\nabla^2 f_k^{-1} \nabla f_k$$

$$p_k = -(J^T J)^{-1} J^T f_k$$

- Classical Gauss-Newton method uses step length $\alpha = 1$ at each iteration

Choosing a search direction: conjugate gradients methods

- Originally developed as an iterative method for solving linear systems of equations of the form

$$Ax = b$$

- Or equivalently minimising convex quadratic functions of the form

$$\phi(x) = \tfrac{1}{2} x^T A x - b^T x$$

- The gradient of φ is equal to the residual of the linear system, i.e.

$$\nabla \phi(x) = Ax - b = r(x)$$

Interlude: conjugate vectors

- A set of nonzero vectors $\{p_0, p_1, \ldots, p_l\}$ are conjugate with respect to the symmetric positive definite matrix A if

$$p_i^T A p_j = 0, \text{ for all } i \neq j$$

Conjugate gradients methods

- Generate a series of conjugate directions that are linear combinations of the steepest descent direction $-\nabla\Phi(x_k)$ and the previous direction p_{k-1} are generated according to

$$p_k = -\nabla f_k + \beta_k p_{k+1}$$

where β_k is given by $$\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$$

- Step length, α_k, is the exact one-dimensional minimiser of quadratic function $\Phi(x_k+\alpha_k p_k)$

$$\alpha_k = -\frac{r_k^T r_k}{p_k^T A p_k}$$

Conjugate gradients methods

- Fletcher and Reeves proposed a generalisation of the conjugate gradient method to minimise nonlinear functions
- Perform a line search to identify approximate minimum of the nonlinear function along p_k

$$\beta_k^{FR} = \frac{\nabla f_{k+1}^T \nabla f_{k+1}}{\nabla f_k^T \nabla f_k}$$