ARP/wARP 8.0

User Guide

1 GENERAL INFORMATION 3

1.1 INTRODUCTION 3
1.2 MAJOR CHANGES IN VERSION 8.0 4
1.3 LATEST NEWS, BUG REPORTS AND TROUBLESHOOTING 5
1.4 DISTRIBUTION 5

2 INSTALLING *ARP/WARP* 6

2.1 STANDALONE INTEL MAC OSX INSTALLATION 6
2.2 STANDALONE COMMAND LINE INSTALLATION ON MAC OSX OR LINUX 6
2.2.1 INSTALLING FOR MULTIPLE USERS 7
2.3 INSTALLATION OF *ARP/WARP*-CCP4 AS A BUNDLE 7

3 USING ARP/WARP 7

3.1 ARP/WARP USING CCP4I 7
3.2 ARP/WARP USING CCP4I2 8
3.3 ARP/WARP WEBSERVICE 8
3.3.1 CREATING A USER ACCOUNT 9
3.3.2 DISSEMINATION LEVEL 9
3.3.3 SUBMITTING ARP/WARP JOBS FROM WEBSERVICE 10
3.4 ARP/WARP USING COMMAND LINE SCRIPTS 10
3.4.1 AUTOMATED MODEL BUILDING WITH `AUTO_TRACING.SH` 10
3.4.2 AUTOMATED CONSTRUCTION OF HELICAL AND BETA-STRANDED FRAGMENTS WITH `AUTO_ALBE.SH` 12
3.4.3 AUTOMATED BUILDING OF POLY-NUCLEOTIDES USING `AUTO_NUCE.SH` 13
3.4.4 AUTOMATED BUILDING OF ATOMIC MODELS INTO CRYO-EM MAPS 13
3.4.4.1 Automated model building and refining as defined in the input sequence file 15
3.4.4.2 Model completion (sequence docking and loop building) 15
3.4.4.3 Hybrid real-reciprocal-space refinement of a model in map 15
3.4.4.4 Cryo-EM map processing. Input map will be scaled in reciprocal space to remove excess sharpening or blurring. 15
3.4.5 AUTOMATED LIGAND BUILDING `AUTO_LIGAND.SH` 16
3.4.6 AUTOMATED SOLVENT BUILDING `AUTO_SOLVENT.SH` 17

4 ACCEPTED FORMATS OF PROTEIN SEQUENCE 18

4.1 EXAMPLE MONOMER: 18
4.2 EXAMPLE HOMO-MULTIMERS: 19
4.3 EXAMPLE MONOMERS OF HETEROMERS: 19
4.4 EXAMPLE HOMO-MULTIMERS OF HETEROMERS: 20

5 QUALITY OF THE X-RAY DATA FOR PROTEIN MODEL BUILDING 20

6 CITING ARP/WARP 21

7 ACKNOWLEDGEMENTS 21

7.1 THIRD PARTY SOFTWARE 22

1 General information

1.1 Introduction

ARP/wARP is a software project for automated protein model building and refinement. *ARP/wARP* combines pattern recognition-based interpretation of an electron density, its modelling as a hybrid model and a maximum likelihood parameter refinement with REFMAC.

The *ARP/wARP* software is under continuous development. Its present release, version 8.0, can be used for the following tasks:

1. Automated protein chain tracing in the density map and model building (CCP4i module *ARP/wARP Classic* and command line module *auto_tracing.sh*). This constructs polypeptide fragments for the cases of MR solutions or MAD/M(S)IR(AS) phases. Generally, the higher the resolution of the X-ray data, the more complete and accurate model *ARP/wARP* will deliver. Typically, X-ray data to 2.7 Å resolution or better are required, although a considerable part of a protein model can sometimes be built at a resolution of 3.5 Å or worse.

2. Automated building of alpha-helical and beta-stranded fragments (CCP4i module *ARP/wARP Quick Fold*, command line module *auto_albe.sh*). This constructs helical and beta-stranded polypeptide fragments (main chain and CB atoms) in low-resolution density maps. Phased X-ray data to 4.5 Å resolution or better are required. This module is automatically invoked as part of protein chain tracing (#1 above) when the resolution of the data is 2.7 Å or worse.

3. Building poorly defined loops in a protein model (CCP4i module *ARP/wARP Loops*). This will generate a set of candidate loops for a short stretch of missing residues given the anchors and the sequence of the missing residues. A protein model and X-ray data to 3.0 Å resolution or higher are required. This module is automatically invoked as part of protein chain tracing (#1 above), provided that the built protein model is sufficiently complete.

4. Software for building poly-nucleotide fragments, DNA or RNA (CCP4i module *ARP/wARP DNA/RNA*, command line module *auto_nuce.sh*). This will produce a set of poly-nucleotide chains with guessed bases (A or C, i.e. large or small), the nucleotide sequence is not yet used. Phased X-ray data to about 3.0 Å resolution or better are required.

5. Automated building of atomic model into cryo electron microscopy maps (CCP4i module *ARP/wARP Classic EM* command line module *auto_em.sh*). The module can automatically build models of proteins, nucleic acids and complexes into maps at 4.5Å resolution or better. Additionally, it can be used for a hybrid real-reciprocal space model refinement, sequence docking and model completion.

6. Building bound ligands (CCP4i module *ARP/wARP Ligands*, command line module *auto_ligand.sh*). This constructs a ligand in a difference electron density map, after the protein model has been completed and refined. It can be given a template search ligand, a list of putative ligands (cocktail screening) or it can guess a ligand among the most common ligands in the Protein Data Bank. X-ray data to 3.0 Å resolution or better are required.

7. Building the solvent structure (CCP4i module *ARP/wARP Solvent*, command line module *auto_solvent.sh*). This builds a solvent structure after the protein model has been refined. The procedure is iterative and uses REFMAC for structure refinement. X-ray data to 3.0 Å resolution or better are required.

8. The remote usage of all ARP/wARP functionalities is possible using ARP/wARP webservice. A submission of crystallographic protein model building (#1 above) to the webservice is also possible from the CCP4i *ARP/wARP Classic* module. Additionally, the webservice provides direct links to ViCi, Auto-Rickshaw and MR pipelines. It can be accessible at https://arpwarp.embl-hamburg.de/

9. A molecular graphics *ARP/wARP* front-end, which allows the display of molecules and electron densities (CCP4i module *ARP Navigator*, executable program *arpnavigator*). It is a high-quality 3D molecular viewer and a user-friendly interface to many *ARP/wARP* functionalities, allowing macromolecular models, secondary structure elements, skeletons, ligands and solvents to be viewed as they are built.

1.2 Major changes in Version 8.0

ARP/wARP 8.0 provides improved performance at resolutions within 2.0-3.5 Å for X-ray crystallography and it is now capable of interpreting cryo-EM maps.

Specifically:
General

- Checks for software installation has been enhanced
- 32 and 64-bit platforms on OSX and Linux are supported
- ARP/wARP is fully compatible with the latest versions of CCP4 and Refmac

Protein model building

- DipCheck conformational space is introduced for the selection of the chain path
- NCS restraints are now default at resolution 1.5 Å or lower
- Loop closure is now provided using a new module Loopy2018

Building models into cryoEM maps

- Protein and protein-nucleic acid models are built with a minimum user input
- Map over-sharpening is handled automatically
- Partial models can be completed by sequence-docking and loop building
- Models can be refined with a combined real- and reciprocal-space protocol

- A very fast secondary structure modelling option can be used separately.

Protein chain tracing
- Chain tracing in both X-ray and cryo-EM maps has been made faster and its memory handling has been improved

Sequence docking and building
- Side chain guess and fit can now be accomplished using a new module SEQQY
- Handling of large structures has been enhanced and the maximum number of NCS copies increased to 60
- Both mono- and hetero-multimers can now be handled

DNA/RNA building
- Nucleotide chain tracing is enhanced with the addition of the second algorithm for phosphate detection
- Nucleotide chain tracing now proceeds in an iterative manner, as for protein tracing

Identification of crystallographic ligands
- Energy estimate for protein-ligand bound state is introduced during identification of crystallographic ligands

Solvent building
- Building of solvent structure exploits advanced features of Refmac including TLS and local NCS restraints. Solvent update can optionally be turned off providing a tool for model refinement with Refmac.

1.3 Latest News, Bug Reports and Troubleshooting

For the latest news and announcements please visit the *ARP/wARP* page (www.arp-warp.org). Some problems and tips can be found on the Frequently Asked Questions link. The developers will greatly appreciate receiving bug reports or suggested changes.

1.4 Distribution

The *ARP/wARP* package (either for download or for remote execution of macromolecule and small molecule model building) is freely available to academic users provided that they agree to the *ARP/wARP* license conditions.

Industrial users are requested to obtain a commercial license via the ARP/wARP *web page.*

Please cite the applications of *ARP/wARP*. Please consult the *ARP/wARP* log file for most relevant citations and also the ARP/wARP web page for the full list of citations.

2 Installing *ARP/wARP*

The recommended way to obtain and install *ARP/wARP* is to use the CCP4 Package Manager, which allows to install CCP4 and ARP/wARP as a bundle, available on the CCP4 download site at `http://www.ccp4.ac.uk/download`. Users can also obtain and install a standalone version of *ARP/wARP*, either from `http://www.arp-warp.org/` or from the CCP4 download site.

ARP/wARP 8.0 supports CCP4 7.0 only. It may work with older versions of CCP4, but the latest features of ARP/wARP 8.0 were not tested with those.

2.1 Standalone Intel Mac OSX Installation

Unless installing a joint CCP4 - *ARP/wARP* bundle, CCP4 must be installed before *ARP/wARP*.
Superuser permissions may be required.
Download arpwarp_8.0.dmg.
Double click on the downloaded file.
Double click on the ARPwARP installer.
Agree to the *ARP/wARP* license.
Select a destination drive.
Choose destination directory if the default /Applications is not desired.
In case of any problems with the installation, we encourage you to save the installation log that is displayed and contact the *ARP/wARP* developers using the link on the *ARP/wARP* homepage.

2.2 Standalone Command Line Installation on Mac OSX or Linux

Unless installing a joint CCP4 - *ARP/wARP* bundle, CCP4 must be installed before *ARP/wARP*.

Download the full *ARP/wARP* package arp_warp_8.0.tar.gz from the CCP4 or *ARP/wARP* web site and save it in a location of your choice. Next, execute:

```
% gunzip arp_warp_8.0.tar.gz
% tar xvf arp_warp_8.0.tar
```

The package will be unpacked under the directory called arp_warp_8.0 that will contain all the required files and subdirectories.
Go to the directory arp_warp_8.0 and run there the install.sh script by simply typing

```
% ./install.sh
```

Superuser permissions may be required. After the installation, restart the CCP4i interface – it should have its model building menu updated.

Example files are provided in the directory arp_warp_8.0/examples for new users to get familiar with ARP/wARP. The included README files give further information about which data are to be used for what purposes.

2.2.1 Installing for Multiple users

Installation for multiple users should normally be the same as for an individual user. Superuser permissions may be required.

2.3 Installation of *ARP/wARP*-CCP4 as a bundle

ARP/wARP can be installed together with CCP4 by downloading and running the package manager available on the CCP4 website.

3 Using ARP/wARP

The ARP/wARP model building jobs can be submitted currently in several different ways:

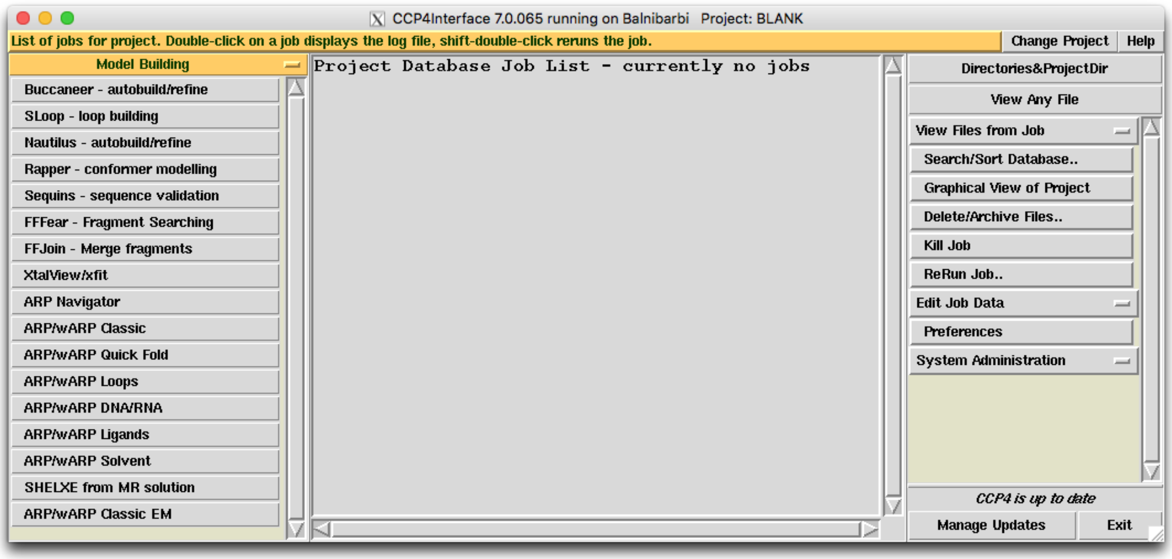
1) from CCP4i interface

2) from CCP4i2 interface

3) via ARP/wARP webservice

4) by using command line scripts

3.1 ARP/wARP using CCP4i



CCP4Interface 7.0.065 running on Balnibarbi Project: BLANK

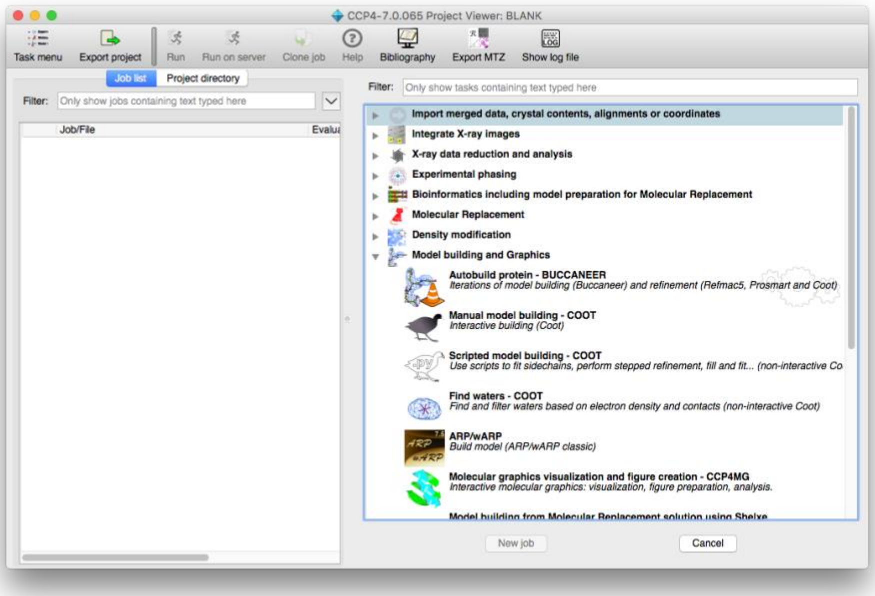
List of jobs for project. Double-click on a job displays the log file, shift-double-click reruns the job.

```
Project Database Job List - currently no jobs
```

CCP4 is up to date

The ARP/wARP modules in the CCP4i and can be viewed as a part of 'Model Building' menu. Their use should be intuitive.

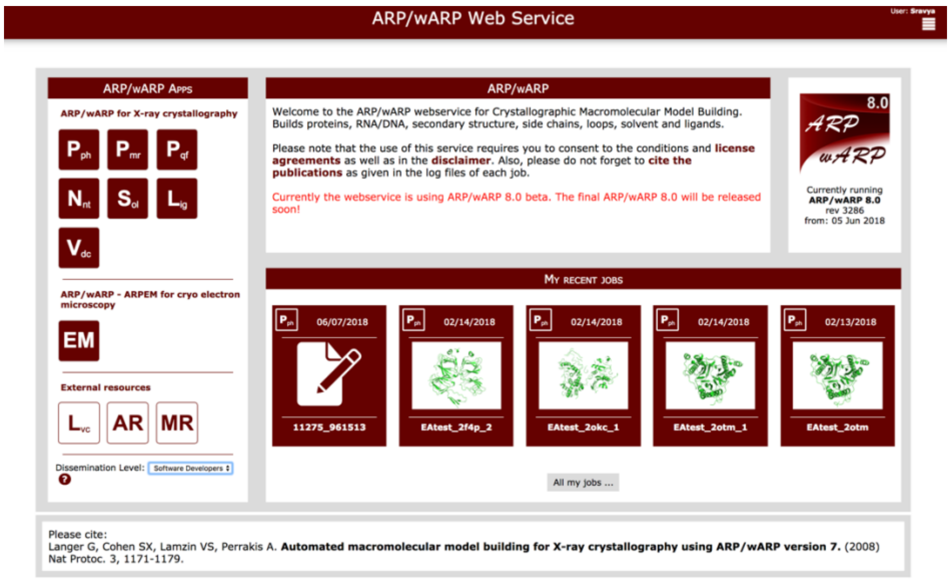
3.2 ARP/wARP using CCP4i2


CCP4-7.0.065 Project Viewer: BLANK
Task menu
Export project
Run
Run on server
Clone job
Help
Bibliography
Export MTZ
Show log file
Job list
Project directory
Filter:
Only show jobs containing text typed here
Job/File
Evalu
Filter:
Only show tasks containing text typed here
Import merged data, crystal contents, alignments or coordinates
Integrate X-ray images
X-ray data reduction and analysis
Experimental phasing
Bioinformatics including model preparation for Molecular Replacement
Molecular Replacement
Density modification
Model building and Graphics
Autobuild protein - BUCCANEER
Iterations of model building (Buccaneer) and refinement (Refmac5, Prosmart and Coot)
Manual model building - COOT
Interactive building (Coot)
Scripted model building - COOT
Use scripts to fit sidechains, perform stepped refinement, fill and fit... (non-interactive Co
Find waters - COOT
Find and filter waters based on electron density and contacts (non-interactive Coot)
ARP/wARP
Build model (ARP/wARP classic)
Molecular graphics visualization and figure creation - CCP4MG
Interactive molecular graphics: visualization, figure preparation, analysis.
Model building from Molecular Replacement solution using Shelxe
New job
Cancel

ARP/wARP model building can also be submitted from CCP4i2. The functionality of ARP/wARP in CCP4i2 may be limited as it is under development.

3.3 ARP/wARP webservice

The *ARP/wARP* webservice is an intuitive web interface for model building, predominantly in macromolecular crystallography, but also in cryo-EM density maps. The webservice supports all *ARP/wARP* modules and provides direct links to DipCheck, ViCi, AutoRickshaw and molecular replacement pipelines Nalbes, Morda and MrBump.


ARP/wARP Web Service
User: Sravya
ARP/wARP Apps
ARP/wARP for X-ray crystallography
P_{ph} P_{mr} P_{qf}
N_{nt} S_{ol} L_{ig}
V_{dc}
ARP/wARP - ARPEM for cryo electron microscopy
EM
External resources
L_{vc} AR MR
ARP/wARP
8.0 ARP wARP
My recent jobs
P_{ph} 06/07/2018 11275_961513
P_{ph} 02/14/2018 EAtest_2f4p_2
P_{ph} 02/14/2018 EAtest_2okc_1
P_{ph} 02/14/2018 EAtest_2otm_1
P_{ph} 02/13/2018 EAtest_2otm
All my jobs ...

Dissemination Level: Software Developers

Welcome to the ARP/wARP webservice for Crystallographic Macromolecular Model Building. Builds proteins, RNA/DNA, secondary structure, side chains, loops, solvent and ligands.

Please note that the use of this service requires you to consent to the conditions and **license agreements** as well as in the **disclaimer**. Also, please do not forget to **cite the publications** as given in the log files of each job.

Currently the webservice is using ARP/wARP 8.0 beta. The final ARP/wARP 8.0 will be released soon!

Currently running **ARP/wARP 8.0** rev 3286 from: 05 Jun 2018

Please cite:
Langer G, Cohen SX, Lamzin VS, Perrakis A. **Automated macromolecular model building for X-ray crystallography using ARP/wARP version 7.** (2008) Nat Protoc. 3, 1171-1179.

Apart from the regular functionalities, the *ARP/wARP* webservice offers following possibilities:

1. Model building will run using external computational facilities, where the CPU performance may be superior to your local installation.
2. Computational tasks use the most recent working executables will be used, should you have a problem with your local installation.
3. Should the task terminate prematurely, an automatic notification will be forwarded to the *ARP/wARP* developers who can then promptly help you.
4. Different dissemination levels can be chosen.
5. The users can view all their jobs, compare them and resubmit them with modified parameters.

3.3.1 Creating a user account

The *ARP/wARP* webserive is accessible at https://arpwarp.embl-hamburg.de/. An account can be created by providing username and email address.

3.3.2 Dissemination level

This defined the level of confidentiality of the data submitted to *ARP/wARP* webservice. This can be selected by using an option 'Dissemination Level' in the *ARP/wARP* Apps tab of the web interface.

Dissemination level:

World This option allows the developers to share the input data provided by the users.

Software Developers This default option allows the input data provided by the users to be shared only with software developers. The advantage of selecting this option is a user can get an assistance from developers in troubleshooting failed model building jobs.

Confidential This option does not allow to store the copy of input data provided by the user.

3.3.3 Submitting ARP/wARP jobs from webservice

After uploading input files to the server, the parameters section will be activated. The job can be launched by clicking ‘Start ARP/wARP’ button. The results page provides the user with real-time details of the computation including the development of R-factor, the number of residues built, Wilson plot data statistics and a 3D view of the built model.

After completion of the task, the results can be downloaded as individual files or as a combined tar file from the ‘Downloads’ tab of the results page. The log file can also be viewed online by clicking ‘show full text log-file’ button at the bottom of the result page.

3.4 ARP/wARP using command line scripts

3.4.1 Automated Model Building with `auto_tracing.sh`

The script auto_tracing.sh in the `$warpbin` directory allows running the automated model building from the command line starting from experimental phases or an existing model (molecular replacement), the so-called *warpNtrace* protocol. The use of auto_tracing.sh is fairly intuitive. If invoked without arguments the script will print help information.

```
Usage:
auto_tracing.sh \
     datafile {mtzfile} \
     [residues {number_of_residues_in_AU}] \
     [workdir {FULLPATH_WORKING_DIRECTORY}] \
     [jobid {name_of_working_subdirectory_and_fileprefix, \
       default is YYYYMMDD_HHMMSS and no file prefix}] \
     [fp {fp_label}] [sigfp {sigfp_label}] [freelabin {freer_label}] \
     [fbest {weighted_amplitude_label}] [phibest {phibest_label}] [fom {fom_label}] \
     [modelin {input_PDB_file_to_use_as_initial_model}] \
     [seqin {sequence_file_for_one_NCS_copy}] \
     [cgr {number_of_NCS_copies (if seqin is provided, default is 1) }] \
     [buildingcycles {the_number_of_autobuilding_cycles (default is 10) }] \
     [resol {'rmin rmax' (default is the full resolution range) }] \
     [albe {1 to always invoke albe, default is 0 for resol < 2.7A, else 1) }] \
     [restraints {1 to use conditional restraints, default is 1 }] \
     [twin {1 to try de-twining and twin refinement, default is 0 }] \
     [compareto {PDB_file_for_comparison}] \
     [keepjunk {1 to keep intermediate models, default is 0 ] \
     [parfile {parfilename_if_only_parfile_is_to_be_created}]

 - Optional command line arguments are given in square parentheses
 - Possible combinations of MTZ labels are:
     For start from phases:
       fp/sigfp/phibest/fom or fbest/sigfp/phibest to build initial free-atoms model
       and fp/sigfp to refine the model
       If 'fbest' is given, 'fom' will be ignored
     For start from a model:
       fp/sigfp to refine the model

 - To carry out SAD refinement in Refmac please provide in addition:
   - phaselabin in a form of ' F+=F+ SIGF+=SIGF+ F-=F- SIGF-=SIGF- '
   - sadcard in a form of ' ANOM FORM SE -7.0 6.5 ' or ' ANOM WAVE 0.98 '
   - heavyin PDB_FILENAME with heavy atom coordinates

 - To carry out phase-restrained Refmac refinement please provide in addition:
   phaselabin in a form of either ' HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM '
   or ' PHIB=PHIM FOM=FOMM ', together with keyword phaseref ' PHAS SCBL 1.0 '

 - All input files are assumed to be located in working directory
   unless they are given with full path
 - If workdir is not given, the current directory will be assumed
 - All output files will be written into workdir/subdirectory


Additional useful tips:
 - Normally the job runs in a subdirectory called YYYYMMDD_HHMMSS
   To run the job in the current directory use: auto_tracing.sh jobId '.'
   To run the job in the directory 'myjob' use: auto_tracing.sh jobId myjob
 - If you have a par file from an earlier version of ARP/wARP and would like to
   re-run that job now, use: auto_tracing.sh defaults OLD_PAR_FILE
   This will create a par file compatible with the current ARP/wARP version
   and the keywords, which are new to OLD_PAR_FILE will take their default values
 - If you invoke auto_tracing.sh from another script and the keywords with
   many-word argument are not properly understood, e.g. resol '20.0 2.5',
   try resol 20.0;2.5 or resol '20.0;2.5'
 - NCS-based chain extension and NCS restraints with Refmac are applied
   automatically if the resolution of the data is equal to or lower than 1.5 A.
   Input 'ncsextension 1/0' to apply / not apply NCS extension regardless of the
   resolution of the data. Input 'ncsrestraints 1/0' has similar effect
```

If auto_tracing.sh is called without an option `parfile`, it will also launch the job. The log files and additional output files as well as the building results can be found in the directory created.

Running model building for the provided example in `arp_warp_8.0/examples/Tracing/PSP/` can be done as follows:

```
% $warpbin/auto_tracing.sh datafile psp.mtz seqin psp.pir fp FP sigfp SIGFP
phib PHIDM fom FOMDM jobid example-psp
```

If *auto_tracing.sh* script is invoked with an option `parfile`, the script will create a parameter file and a directory whose names will be printed. The job can subsequently be launched by:

```
% $warpbin/warp_tracing.sh NAME_OF_PARFILE
```

3.4.2 Automated Construction of Helical and Beta-Stranded Fragments with `auto_albe.sh`

The script *auto_albe.sh* (where 'albe' stands for alpha-beta) should be able to build helices and strands at resolutions as low as 4.5 Å. However, it may not result in complete helical/stranded structure. The expected top performance is the correct location of 90% of the helices and 50% of the strands. The procedure is relatively fast and takes only seconds to minutes for proteins of moderate size (up to 500 residues). The use of *auto_albe.sh* is fairly intuitive. If invoked without arguments the script will print help information.

```
Usage:
auto_albe.sh \
     datafile {mtzfile}  \
     [residues {number_of_residues_in_AU}] \
     [workdir {FULLPATH_WORKING_DIRECTORY}] \
     [helixfileout {output_PDB_file}] \
     [jobid {desired_job_id_used_for_subdirectory_naming}] \
     [fp {fp label} sigfp {sigfp label} phib {phi label}] \
     [fom {fom label}] (input 'fom none' if no fom is to be used) \
     [compareto {PDB_file_for_comparison}] \
     [nostrands {0 or 1, default=0}] \
     [parfile {parfilename_if_only_parfile_is_to_be_created}]

 - Optional command line arguments are given in square parentheses
 - All input files are assumed to be located in working directory
   unless they are given with full path
 - If workdir is not given, the current directory will be assumed
 - All output files will be written into workdir/subdirectory
```

If auto_albe.sh is called without an option `parfile`, it will also launch the job. The log files and additional output files as well as the building results can be found in the directory created.

Running secondary structure building for the example provided example in `arp_warp_8.0/examples/Secstruct/1o14/` can be done as follows:

```
% $warpbin/auto_albe.sh datafile combined_dm1.mtz fp F sigfp SIGF phib PHIC
fom FOM jobid example-1o14
```

If *auto_albe.sh* script is invoked with an option `parfile`, the script will create a parameter file and a directory whose names will be printed. The job can subsequently be launched by:

```
% $warpbin/warp_tracing.sh NAME_OF_PARFILE
```

3.4.3 Automated Building of Poly-Nucleotides using `auto_nuce.sh`

The script *auto_nuce.sh* can be used to build poly-nucleotide fragments. The procedure is relatively fast and normally takes only a few minutes. The use of *auto_nuce.sh* is fairly intuitive. The script prints out help information if it is invoked without arguments.

```
Usage:
auto_nuce.sh \
     datafile {mtzfile} \
     [residues {number_of_protein_residues_in_AU}] \
     [nucleotides {number_of_nucleotides_in_AU}] \
     [workdir {FULLPATH_WORKING_DIRECTORY}] \
     [jobid {desired_job_id_used_for_subdirectory_naming}] \
     [fp {fp_label}] [sigfp {sigfp_label}] [fbest {weighted_amplitude_label}] \
     [phib {phib_label}] [fom {fom_label}] \
     [resol {'rmin rmax' (default is the full resolution range) }] \
     [compareto {PDB_file_for_comparison}] \
     [parfile {parfilename_if_only_parfile_is_to_be_created}] \

 - Optional command line arguments are given in square parentheses
 - Possible combinations of MTZ labels for map calculation are:
     fp/sigfp/phib/fom or
     fbest/sigfp/phib if fbest is already fom-weighted.
 - In the latter case if 'fbest' is given, 'fom' will be ignored

 - All input files are assumed to be located in working directory
   unless they are given with full path
 - If workdir is not given, the current directory will be assumed
 - All output files will be written into workdir/subdirectory
```

If auto_nuce.sh is called without an option `parfile`, it will also launch the job. The log files and additional output files as well as the building results can be found in the directory created.

Running nucleotide building for the example provided example in
`arp_warp_8.0/examples/ examples/DNA/` can be done as follows:

```
% $warpbin/auto_nuce.sh datafile combined_dm1.mtz fp F_eh2nm sigfp SIGF_eh2nm
phib PHIDM fom FOMDM jobid example-muts
```

If *auto_nuce.sh* script is invoked with an option `parfile`, the script will create a parameter file and a directory whose names will be printed. The job can subsequently be launched by:

```
% $warpbin/warp_nuce.sh NAME_OF_PARFILE
```

3.4.4 Automated building of atomic models into cryo-EM maps

The script *auto-em.sh* can be used for building and refinement of atomic models into cryo-EM maps at resolution 4.5Å or better. It handles models of proteins, nucleic acids, and their complexes fully automatically. There are four basic functionalities provided by the script: automated model building, model refinement, model completion, and map pre-processing. The script prints out help information if it is invoked without arguments.

```
Usage: auto_em.sh [options]

This is the ARP/wARP module for building protein, nucleic-acid, and protein-nucleic
acid complex models into cryo-EM maps.
The method works best at resolutions when side-chains start to be visible, that is
usually up to around 3.5A.



Options:
  -h, --help            show this help message and exit

  Modus operandi:
    --refine            Real space refinement followed with several Refmac
                        runs to adjust B-factors and fine-tune coordinates.
                        Required parameters: mapin, modelin, resamx. Number of
                        Refmac cycles may be adjusted wirh rrcyc.
    --build             Build a model into input EM map. By default
                        macromolecule type will be guessed form input
                        sequence, but you may also force it with --chain_type.
                        Required parameters: mapin, seqin, resmax
    --process           Adjusts sharpening/blurring of a map, no model
                        building. By default, the sharpening/blurring factor
                        is determined automatically, but you can force a
                        specific value with --badd.
    --loops             Build missing loops. Required parameters: modelin,
                        mapin, and seqin

  Standard parameters:
    --mapin=FILENAME    input map in CCP4 format
    --seqin=FILENAME    Target model sequence. Current limit for the total
                        number of protein residues is 12000.
    --cgr=VALUE         Number of entities defined in seqin to be build. If
                        seqin is complete (all chain sequences provided) use
                        default cgr=1
    --chain_type=[auto,prot,nuce,cplx]
                        Select chain type to be automatically build (protein,
                        nucleic acid, or protein-nucleic acid complex).
                        Default [auto] will try to guess it from input
                        sequence
    --prot_residues=VALUE
                        Total number of protein residues in the target
                        structure.
    --nuce_residues=VALUE
                        Total number of RNA/DNA residues in the target
                        structure.
    --resmax=VALUE      Input map resolution [default: none]
    --modelin=FILENAME  Input model
    --albe              Build alpha/beta protein only. There will be no
                        sequence docking step (you can do it later with
                        --loops) but you will get a nice model in no time
    --dock              Dock sequence before loop building (loop mode only).
                        Otherise input model sequence will be used to
                        determine missing fragments.

  Optional parameters:
    --workdir=DIRNAME   Project working directory name [default: cwd]
    --jobid=DIRNAME     Name of working subdirectory, based on mapin by
                        default
    -e, --expert        show expert options
    -x, --examples      show command-line examples and exit

  Expert parameters:
    Changes may have unpredictable consequences. Defaults should provide
    best results in most of the cases. Enable with -e/--expert
```

Explicit example commands that use test-data provided with ARP/wARP can be displayed with a command:

```
% auto_em.sh -x
```

3.4.4.1 Automated model building and refining as defined in the input sequence file

```
   % auto_em.sh --mapin [map filename] --seqin [sequence filename] --resmax [res]
      --build
```

The program will build and refine a model defined in the input sequence file. Required options are input map, estimated map resolution and sequence. Number of copies of a molecule defined in the input sequence can be changed with `--cgr`. Additionally, a very fast and accurate tracing of protein secondary structure elements only may be triggered with an option --`albe` (albe stands for alpha/beta). This option is used by default at resolutions below 4.5 Å, but you may consider using it at lower resolution for initial, very fast evaluation of a map quality and/or resolution. When --`albe` option is used one can specify number of residues in target molecule with --`prot_residues` keyword instead a complete sequence.

3.4.4.2 Model completion (sequence docking and loop building)

```
   % auto_em.sh --mapin [map] --seqin [sequence] --resmax [res] --model [input model]
      --loops
```

The program will automatically determine and try to build all missing loops in the input model based on the input sequence. Additionally, sequence docking prior loop building can be triggered with a --`dock` option.

3.4.4.3 Hybrid real-reciprocal-space refinement of a model in map

```
   % auto_em.sh --mapin [map] --resmax [res] --model [input model] --refine
```

By default, the program will perform real-space refinement with ARP/wARP tools and 100 reciprocal-space refinement cycles with Refmac. The number of Refmac cycles can be changes with an option --`rrcyc`.

3.4.4.4 Cryo-EM map processing. Input map will be scaled in reciprocal space to remove excess sharpening or blurring.

```
   % auto_em.sh --mapin [map] --resmax [res] --process
```

By default, the program will determine an optimal, global Wilson-B factor for a given resolution and scale the map accordingly. A specific sharpening factor (change in Wilson-B) can be set with --`badd` keyword.

3.4.5 Automated Ligand Building `auto_ligand.sh`

The script *auto_ligand.sh* can be used to build/guess small-molecule ligands in crystallographic maps. The procedure is relatively fast and normally takes only a few minutes. The use of *auto_ligand.sh* is fairly intuitive. The script prints out help information if it is invoked without arguments.

```
Usage:
auto_ligand.sh                                                          \
  datafile {either mtzfile or mapfile}                                  \
  protein {starting_PDB_file_without_ligand}                            \
  [ligand {PDB_file_with_ligand_to_fit}]                                \
  [ligandcode {3-letter code of a ligand molecule,                      \
    the code must be present in the refmac library}]                    \
  [workdir {FULLPATH_WORKING_DIRECTORY}]                                \
  [ligandfileout {output_PDB_file}]                                     \
  [fp {fp_label}] [sigfp {sigfp_label}] [freelabin {free_label}]        \
  [nligandcycles {number_of_ligandbuild_cycles (default is 2)}]         \
  [search_model {PDB_file_with_model_at_expected_ligand_site}]          \
  [search_position 'X Y Z']                                             \
  [search_radius {radius_in_angstroms}]                                 \
  [reflist {textfile_with_FULLPATHnames_of_fitted_ligands_for_comparison}] \
  [extralibrary {user_defined_library_for_Refmac5}, additionally        \
    if this library contains data for the ligand to be built, then these \
    parameters are used to derive ligand topology to the highest level  \
    of priority (ahead of REFMAC cif or coordinate-derived topology)]   \
  [partial {0 for modelling the whole ligand and 4 or higher number to  \
    model partially occupied ligand (giving 4 would mean to consider    \
    4-atoms as the smallest ligand fragment)]                           \
  [keepwaters {1 for keeping them before computing the difference map}] \
  [parfile {parfilename_if_only_parfile_is_to_be_created}]

 - Optional command line arguments are given in square parentheses
 - All input files are assumed to be located in working directory
   unless they are given with full path
 - If workdir is not given, the current directory will be assumed
 - All output files will be written into workdir/subdirectory
 - If no ligand is specified then auto identification of
   the ligand will be attempted provided that a search position
   is given (experimental)


Additional useful tips:
 - Default geometry weight is: weightv 'AUTO 7'
    for tighter geometry change to, e.g. weightv 'AUTO 5'
 - If you invoke auto_ligand.sh from another script and the keywords with
    many-word argument are not properly understood, e.g. resol '10.0 2.0',
    try resol '20.0;;2.5'
```

The *ARP/wARP* ligand building module requires the X-ray data in MTZ format or a density map, the protein without ligands in PDB format and either a template model of the ligand to build (also in PDB format) or a ligand 3-letter code. Options include the possibility to specify the binding site, and the possibility to build a ligand taken from a list of candidates (‘cocktail’). In the latter case, the coordinates of the ligand candidates should be concatenated into a single PDB file. The different ligands must be distinguished by their residue names, chain identifiers or residue numbers. One can also specify that only well-resolved parts of a partially occupied ligand should be modelled and indicate the minimum number of atoms present in the bound ligand fragment.

If auto_ligand.sh is called without an option `parfile`, it will also launch the job. The log files and additional output files as well as the building results can be found in the directory created.

Running ligand building for the provided retinoic acid example in `arp_warp_8.0/examples/Ligands/` can be done as follows:

```
auto_ligand.sh datafile 1cbs.mtz protein 1CBS_noligand.pdb ligand
RETINOICACID.pdb
```

If *auto_ligand.sh* script is invoked with an option `parfile`, the script will create a parameter file and a directory whose names will be printed. The job can subsequently be launched by:

```
% $warpbin/warp_ligand.sh NAME_OF_PARFILE
```

3.4.6 Automated Solvent Building `auto_solvent.sh`

The script *auto_solvent.sh* can be used to build solvent structure around an already complete protein model. The procedure is iterative but is relatively fast and normally takes less than an hour. The use of *auto_solvent.sh* is fairly intuitive. The script prints out help information if it is invoked without arguments.

```
Usage:
$warpbin/auto_solvent.sh \
     datafile {mtzfile} \
     protein {starting_PDB_file} \
     [arpmode {noupdate (default is update, 'arpmode waters') }] \
     [extralibrary {file with a user_defined_CIF_library_for_Refmac5}] \
     [ncsr_local {0 to not use local NCS restraints, default is 1 }] \
     [fp {fp_label}] [sigfp {sigfp_label}] \
     [freelabin {freer_label}] \
     [freename {residue_name_for_solvent_atoms, default WAT }] \
     [restrcyc {maximum_number_of_cycles (default is 20) }] \
     [rrcyc {number_of_internal_refmac_cycles (default is 5) }] \
     [resol {'rmin rmax' (default is the full resolution range) }] \
     [solventfileout {output_PDB_file}] \
     [tlsrefine {1 to do TLS refinement, default is 0 }] \
     [twin {1 to do twin refinement with Refmac, default is 0 }] \
     [workdir {fullpath_working_directory}] \
     [parfile {parfilename_if_only_parfile_is_to_be_created}]

 - Optional command line arguments are given in square parentheses

 - All input files are assumed to be located in working directory
   unless they are given with full path
 - If workdir is not given, the current directory will be assumed
 - All output files will be written into workdir/subdirectory
 - Normally the job runs in a subdirectory called YYYYMMDD_HHMMSS
```

```
   To run the job in the directory 'myjob' use: auto_solvent.sh jobid myjob


Additional useful tips:
 - Default geometry weight is: weightv 'AUTO 7'
    for tighter geometry change to, e.g. weightv 'AUTO 5'
 - If you invoke auto_solvent.sh from another script and the keywords with
    many-word argument are not properly understood, e.g. resol '10.0 2.0',
    try resol '20.0;;2.5'
 - If tlsrefine is 1, the TLS tensor can be computed as follows
    - refined anew (this is a default)
    - be taken from a tls file [tlsin file.tls]
    - be taken from a PDB file [tlsin file.pdb]
```

If auto_solvent.sh is called without an option `parfile`, it will also launch the job. The log files and additional output files as well as the building results can be found in the directory created.

Running solvent building for the example provided example in `arp_warp_8.0/examples/ examples/DNA/` can be done as follows:

```
% $warpbin/auto_ solvent.sh datafile 3raz.mtz protein 3raz_start.pdb fp F
sigfp SIGF jobid example-3raz
```

If *auto_solvent.sh* script is invoked with an option `parfile`, the script will create a parameter file and a directory whose names will be printed. The job can subsequently be launched by:

```
% $warpbin/warp_solvent.sh NAME_OF_PARFILE
```

4 Accepted formats of protein sequence

The protein sequence for ARP/wARP model building should be in a pir/fasta format. Each sequnce fragment starts with a comment line (starting with >) followed by lines containing the amino-acid sequence in single-character code and in free format.

During ARP/wARP sequence check the chain repeats will be interpreted as homo-multimers (NCS or non-crystallographic symmetry). Heteromers will contain additionally introduced strentches of 10 alanine residues which will unlikely to bebuilt in the density map.

Examples of the content of the sequence file are given below.

4.1 Example monomer:

NCS 1, fragments 1, sequence length 25

```
> Title
VLSPADKTNVKAAWGKVGAHAGEYG

or
```

```
VLSPADKTNVKAAWGKVGAHAGEYG

=
or

VLSPADKTNVKAAWGKVG

AHAGEYG

or

>
>
VLSPADKTNVKAAWGKVGAHAGEYG
>
>
>

The cases when the sequence is placed inside the comment line
may also be interpreted correctly although the use of such
format is discouraged:

> VLSPADKTNVKAAWGKVGAHAGEYG
```

4.2 Example homo-multimers:

NCS 2, fragments 1, sequence length 2x25

```
> Title
VLSPADKTNVKAAWGKVGAHAGEYG
  > Title
VLSPADKTNVKAAWGKVGAHAGEYG

or

> Title
VLSPADKTNVKAAWGKVGAHAGEYG
VLSPADKTNVKAAWGKVGAHAGEYG

or

VLSPADKTNVKAAWGKVGAHAGEYGVLSPADKT
NVKAAWGKVGAHAGEYG
```

4.3 Example monomers of heteromers:

NCS 1, fragments 3 (with 10-alanine stretches introduced between them), sequence length 77

```
> Title
VLSPADKTNVKAAWGKVGAHAGEYG
> Title
LRLAIIAELDAINLYEQMARYSE
> Title
RKILLDVAREEKAHVGEFMALLLNLDPEQ
```

```
or

   VLSPADKTNVKAAWGKVGAHAGEYG
> Title
LRLAIIAELDA
INLYEQMARYSE
> Title
RKILLDVAREEKAHVGEFMALLLNLDPEQ
```

4.4 Example homo-multimers of heteromers:

NCS 2, fragments 3 (with 10-alanine stretches introduced between them), sequence length 2x77

```
> Title
VLSPADKTNVKAAWGKVGAHAGEYG
> Title
LRLAIIAELDAINLYEQMARYSE
> Title
RKILLDVAREEKAHVGEFMALLLNLDPEQ
> Title
   VLSPADKTNVKAAWGKVGAHAGEYG
> Title
LRLAIIA
ELDAINLYEQMARYSE
> Title
RKILLDVAREEKAHVGEFMAL
LLNLDPEQ
```

5 Quality of the X-ray Data for protein model building

The space group of the X-ray data should be correctly determined. A use of data in incorrect space group is a frequent cause for *ARP/wARP* to build only 50% (or less) of the model in short fragments.

The X-ray data should be as complete as possible, especially in the low resolution range (10 Å and worse). *ARP/wARP* automatically checks the fit of the X-ray data to the expected Wilson plot, and the user is advised to visually inspect the Wilson plot and apply his/her critical judgement as to whether or not the data should be cut. It has sometimes proved beneficial to cut the low-resolution data beyond 15 or even 10 Å. It may also be advantageous to cut the high-resolution data if their intensity fall-off grossly deviates from the expected Wilson plot..

The Wilson plot can be expected by clicking on the *Wilson.loggraph in the CCP4i job menu, or by executing the following on a command line:

```
% $CBIN/loggraph NAME_OF_WILSON_LOGGRAPH_FILE
```

6 Citing ARP/wARP

Please cite the applications of *ARP/wARP* that you have used. Please consult the *ARP/wARP* log file for the most relevant citation and/or the *ARP/wARP* web page http://arp-warp.org for all ARP/wARP citations..

7 Acknowledgements

The current *ARP/wARP* developers are:
The Hamburg team (European Molecular Biology Laboratory (EMBL) Hamburg, c/o DESY, Notkestrasse 85, 22607 Hamburg, Germany):

- Victor Lamzin
- Grzegorz Chojnowski
- Sravya Mounika Kantamneni
- Umut Deniz Oezugurel
- Egor Sobolev

Former members

- Daria Beshnova, Ciarán Carolan, Serge Cohen, Zbyszek Dauter, Helene Doerksen, Guillaume Evrard, Francisco Fernandez, Johan Hattne, Saul Hazledine, Philipp Heuser, Marouane Jelloul, Krista Joosten, Matheos Kakaris, Olga Kirillova, Gerrit Langer, Wijnand Mooij, Richard Morris, Venkat Parthasarathy, Joana Pereira, Anastassis Perrakis, Tilo Strutz, Ioan Vancea, Tim Wiegels, Keith Wilson, Peter Zwart

The authors are especially grateful to:

- Keith S Wilson (York, UK) one of the originators of the software, Zbyszek Dauter (Argonne, USA) for significant contributions at earlier stages the software development and Eleanor Dodson (York, UK) for continued support of ARP/wARP since early 1990th.
- Anastassis Perrakis (Amsterdam, NL), the former co-developer, for his great input and encouraging spirit.
- The REFMAC developers team lead by Garib Murshudov (York-Cambridge, UK).
- The CCP4 developers currently lead by Eugene Krissinel (Didcot, UK)
- Many of our collaborators and all tens of thousands of ARP/wARP users

We would also like to take this opportunity to thank for the continuing support of *ARP/wARP*: the EMBL for hosting the research group, the *ARP/wARP* download servers and remote computational infrastructure, funding agencies for research and infrastructure grants; and *ARP/wARP* academic and industrial users.

7.1 Third Party Software

The *ARP/wARP* distribution includes an unmodified version of the Open Astex Viewer software (`http://openastexviewer.net/web/license.html`).

The *ARP/wARP* distribution also includes a modified version of the smi23d software (`http://www.chembiogrid.org/cheminfo/smi23d/`) in the form of executables – smi23d and mengine. The smi23d software is covered by the Apache License, Version 2.0 (`http://www.apache.org/licenses/LICENSE-2.0.html`).