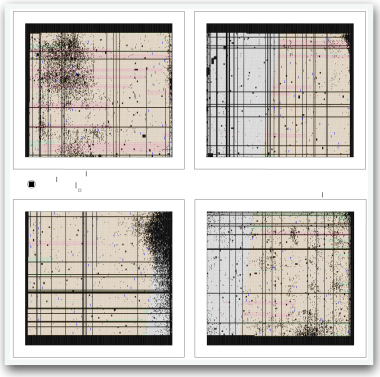
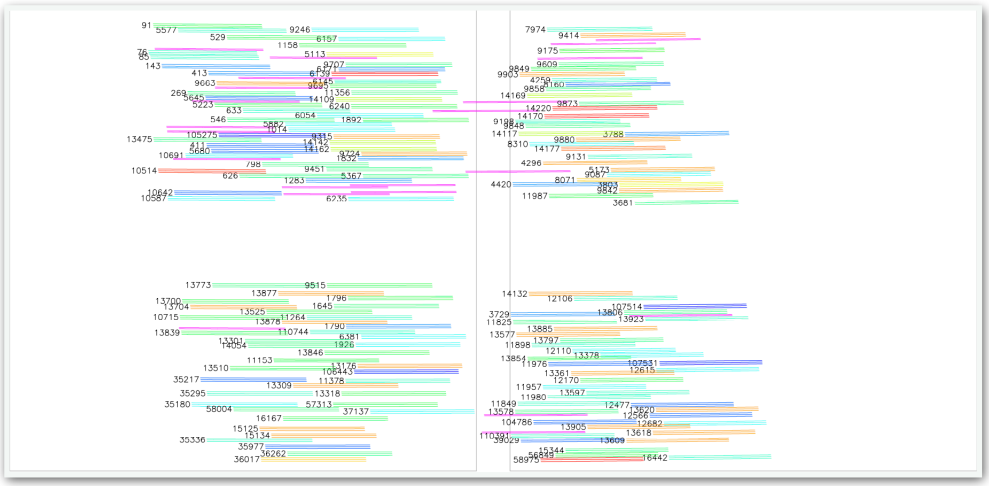
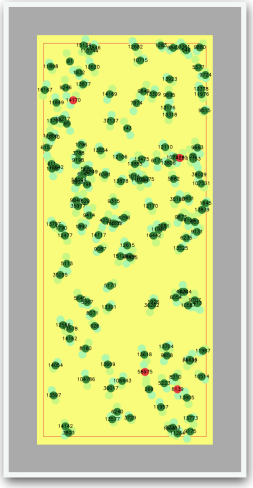
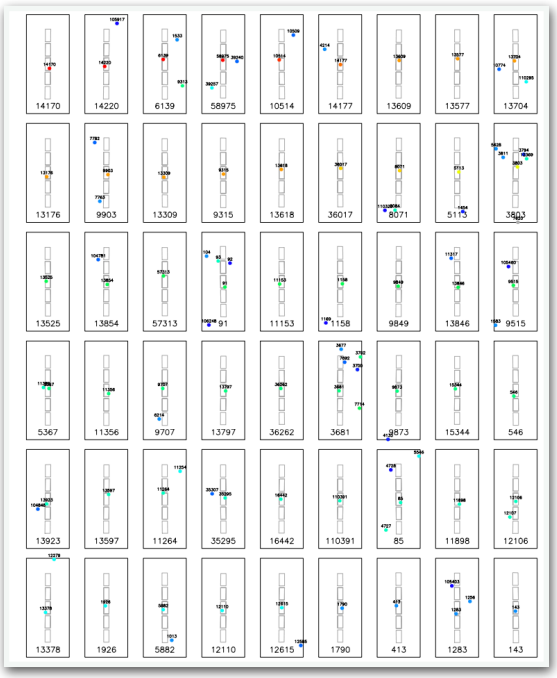
eMPT Software & User Guide

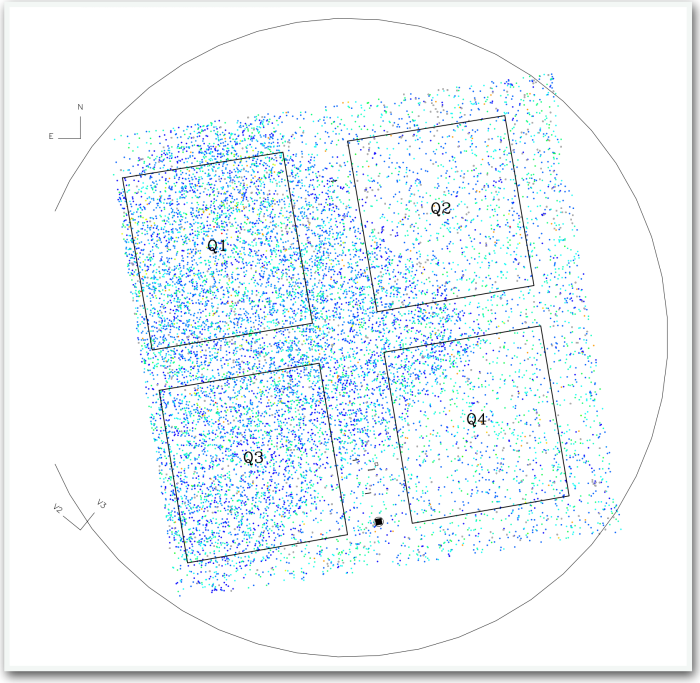


Installation and usage instructions for the eMPT software package, an alternative to the MSA Planning Tool (MPT) in the Space Telescope Science Institute's (STScI) Astronomers' Proposal Tool (APT) for James Webb Space Telescope (JWST) Near-infrared Spectrometer (NIRSpec) Multi-object Spectroscopic (MOS) proposal planning.









Prepared by: Nina Bonaventura & Peter Jakobsen **Issue/Revision**: 1.1 **Date:** 09 May 2023

CONTENTS

1 Introduction & Background

The eMPT software suite is offered to JWST observers as a supplement, or full alternative to, the STScI MPT for designing the most scientifically optimal MSA configurations for their planned NIRSpec MOS observations. The eMPT was developed in collaboration with the NIRSpec Guaranteed Time Observer (GTO) team with sophisticated algorithms and a modular architecture in order to achieve the ambitious goals of its science program. As a result, the eMPT provides a highly flexible and customizable interface for the user, with options to finely control the workflow and insert their own software modules to tune their MSA slit masks to the particular scientific objectives at hand.

This guide contains detailed instructions for the installation, testing, and operation of the software using a test data set that represents a common user case, as well as for the ingestion of the eMPT output into the APT in order to complete a NIRSpec MOS observing proposal using the eMPT-customized MSA configurations.

The eMPT utilizes an up-to-date, in-flight version of the NIRSpec instrument model, ensuring maximal scientific return for a given observing program by carefully prioritizing targets as desired, and maximizing the number of spectra that can cleanly and safely be dispersed together onto the detector arrays in a single exposure. Its algorithms match the ~250,000 individually addressable micro-shutters of the NIRSpec MSA to the targets listed in a user-supplied prioritized input catalog, producing output that can be entered directly into the STScI APT. Compared to the MPT, the eMPT offers additional user options for dither selection and an automatic utilization of spare, empty MSA shutters for a master background calculation. In addition, its fundamentally different search algorithm for identifying optimal pointings ensures that the maximum number of user-specified highest-priority targets are simultaneously observed in the output MSA configuration, for any given input

1.1 Contents of the eMPT

In its present modular form, the eMPT software suite consists of seven independent Fortran modules that are run in strict sequence, the output of which the user can interactively examine and approve, and optionally re-run with modified parameter settings before proceeding to the next step. Alternatively, a full automatic run of the software can be quickly executed from the command line, when only the final output summary, diagnostic, and plot files need to be examined. There is also a user-configurable Python template script available for running the eMPT in batch mode, designed for experienced users who may need to run many exploratory trials in succession with different starting parameters set for each.

The seven eMPT modules are compiled with Fortran90 and depend on the Fortran PGPLOT plotting library for the numerous PostScript informational and diagnostic plot files it generates; it also internally utilizes Python version 3.x for instrument model parameter updates. The user is ready to run the eMPT once assigned a roll angle by the STScI planning system, and wishes to determine the final optimal matching NIRSpec pointing in RA and Dec consistent with the allowed deviation from the preliminary nominal pointing claimed at the program proposal stage. The MSA configuration file and final observed target summary information associated to the optimal single pointing, or set of up to three simultaneously optimized dithered pointings, determined by the software may then be entered into the APT to generate the necessary Visit files for the optimal MSA observation program at hand.

Core Fortran modules

The core Fortran modules comprising the eMPT, named here, are executed from the Unix command line in the following strict sequence: *ipa*, *k_make*, *k_clean*, *m_make, m_sort, m_pick, m_check*. In summary, the task of the *ipa* module is to locate all allowable NIRSpec pointings at a given roll angle that maximize the simultaneous coverage of the highest-priority targets, those categorized as Priority Class 1 (PC1) in the input target catalog (3.1.3.1.a). The following four modules, *k_make*, *k_clean*, *m_make,* and *m_sort,* together then endeavor to identify which of these PC1-optimizing pointings also provide optimal coverage of the remaining lower Priority Class 2+ targets in the input catalog. The final *m_pick* and *m_check* modules then produce the diagnostic information describing the optimal solution in detail, along with the information needed to enter it into the STScI APT MPT. A brief description of the functionality of each module is contained in Section 3.2; a description of the main output files in Section 4.5; and a full listing and associated descriptions of all output files of all modules are provided in the Appendix for reference.

Controlling Configuration File

The eMPT process chain is controlled by a single ASCII-formatted configuration file that is read by each module in turn, and which conveys to the software the options and parameters selected by the user as appropriate to the NIRSpec MSA observation being planned. A template configuration file populated with default parameter values is included in the software package, and was designed to be directly edited by the user.

Input Data and Parameters

User-provided data and parameter values:

The required data items and parameter values that the user must assemble and enter into the

configuration file (Section 3.1.2) in order to run the software, include:

(a) The name of the eMPT-formatted target input catalog listing the identifiers, coordinates, and priority class designations of the objects to be observed.

All targets in the catalog must be assigned a Priority Class. Priority Class 0 is reserved for stars and other foreground objects; Priority Classes > 1 mainly serve to group the targets into similar type of object and need not necessarily reflect the relative scientific importance of each group. The prioritized order in which targets of different Priority Class are attempted observed can be customized by the user separately (see *k_clean* module output description in Section 4.5.2).

The input catalog must adhere to the following format:

Column 1: Integer identifier in the form of the running number of the target in the master catalog from which the eMPT-formatted catalog was extracted.

Column 2: Right Ascension of the target in decimal degrees (double precision real)

Column 3: Declination of the target in decimal degrees (double precision real)

Column 4: Integer Priority Class designation of the target

Any additional columns beyond these four will be ignored by the software.

(**b)** The Right Ascension, Declination, and Roll angle (*pa_ap*) specifying the nominal pointing of the observation in decimal degrees. The specified (double precision real) RA and Dec values are meant to be the provisional ones provided to STScI in the observation proposal stage. The specified (real) roll angle (PA_AP in STScI parlance) should be be the final one assigned by STScI once the observation in question has been scheduled. This roll angle is assumed fixed throughout the eMPT processing chain.

(**c)** The identifying number of the eMPT run, specified by an integer, *id*, between 0 and 99. The various files produced by each module are written to separately named output subdirectories within the overall directory *./trial_id/*. The output of the *ipa* module is therefore placed in *./trial_id/ipa_output/*.

(**d)** Name of the disperser for which the baseline MSA mask yielding non-overlapping spectra is to be produced; options (with associated filter choice) include:

PRISM/CLEAR, G140M/F070LP, G140M/F110LP, G235M/F170LP, G395M/F290LP. For all choices of disperser/filter, the use of slitlets leading to spectra overlapping with those of the failed open shutters are avoided. In the case of the PRISM, the spectra will furthermore be prevented from intercepting the gap between the two arrays making up the NIRSpec detector.

(**e)** The (integer) number of matching optimal "dithered" pointings (*n_dither)* that are to be found by the *ipa* module within the search area specified in (h). This key parameter may only take on the values *n_dither*=1, *n_dither*=2 or *n_dither*=3. The choice of *n_dither*=1 signals that the single pointing providing optimal coverage of Priority Class 2 and higher targets is to be located among the Priority-Class-1-optimizing pointings identified by the *ipa* module. When *n_dither*=2 the subsequent *m_make* will search among the optimal Priority-Class-1-covering pointings for the optimal PAIR of such pointings whose COMBINED observed target list is optimal in a user-determined fashion (see description

of the *m_sort* module). Similarly, when *n_dither*=3, the TRIPLE set of *ipa*-identified pointings whose total combined observed target list is optimal is located.

Note on dithers: The user is presented with options for controlling the manner in which *ipa*-output optimal pointings should be searched and then sorted on output, in the *ipa.conf* configuration file located in the *reference_files* subdirectory of eMPT installation directory (see Section 4.2). Setting the *group_output* parameter in this file to '.true.' (default) will sort the output optimal pointings into groups covering the same sets of targets, primarily intended for cases where *n_dither*=2 or *n_dither*=3. Setting this parameter to '.false.' instead will rank the output optimal pointings per decreasing number of Priority 1 targets covered, primarily intended for cases where *n_dither*=1. Additional *ipa* parameter settings and associated descriptions are conveniently provided for the advanced user in *ipa.conf*.

Pre-packaged default settings:

The remaining parameters that are specified in the configuration file, listed below, remain set at their pre-packaged default values and are recommended for the majority of use cases.

However, when desired, the user may directly edit them using a standard text editor.

(f) Half-extent of the Acceptance Zone in MSA X and Y measured in units of the mean MSA shutter pitch. The default values of 0.353 and 0.425 represent the full shutter open area with 5 millarcseconds (mas) "shaved off" around the edges to ensure that the differential optical distortion does not lead to targets located close to the boundary of the Acceptance Zone in the central pointing falling outside the open area in any of the two nodded pointings.

(g) Minimum vertical spectral separation measured in units of the shutter pitch in Y required for any two spectra to be considered as non-overlapping. This parameter only applies to the baseline disperser specified in (d) above. The default minimum value of 3.5 corresponds to the vertical distance between the centers of two adjacent 3-shutter-tall slitlets with half of one shutter height added for margin. Larger values of this parameter will lead to a better vertical separation of spectra, albeit at the obvious cost of a decrease in the total number of spectra that can be placed on the detector without incurring overlap.

(h) The (real) maximum allowable pointing shift (half-range in MSA X and Y in decimal arcseconds) that the final actual Right Ascension and Declination of the final pointing of the observation is allowed to deviate from the nominal pointing at the fixed roll angle specified in (i-b) above. The default value of +/-25.0 arcseconds in both dimensions corresponds to the maximum range available if the entire MSA field of view is to stay within a 3-arcminute-radius circle surrounding the nominal pointing.

(i) The maximum acceptable contamination scores for Priority Class 1 targets [integer, Default '1 2 2', set to '100 100 100' to disable], here defined by the presence of any other targets in any of the five sky shutters sampled by the baseline three nodded exposures, and still be deemed worthy of being observed. The first of the three integers applies to the shutters nominally containing intended targets, defined by the following contamination scoring scheme:

1: Intended target in shutter only
4: One non-target object allowed in shutter
8: Two non-target objects allowed in shutter
16: Three non-target objects allowed in shutter

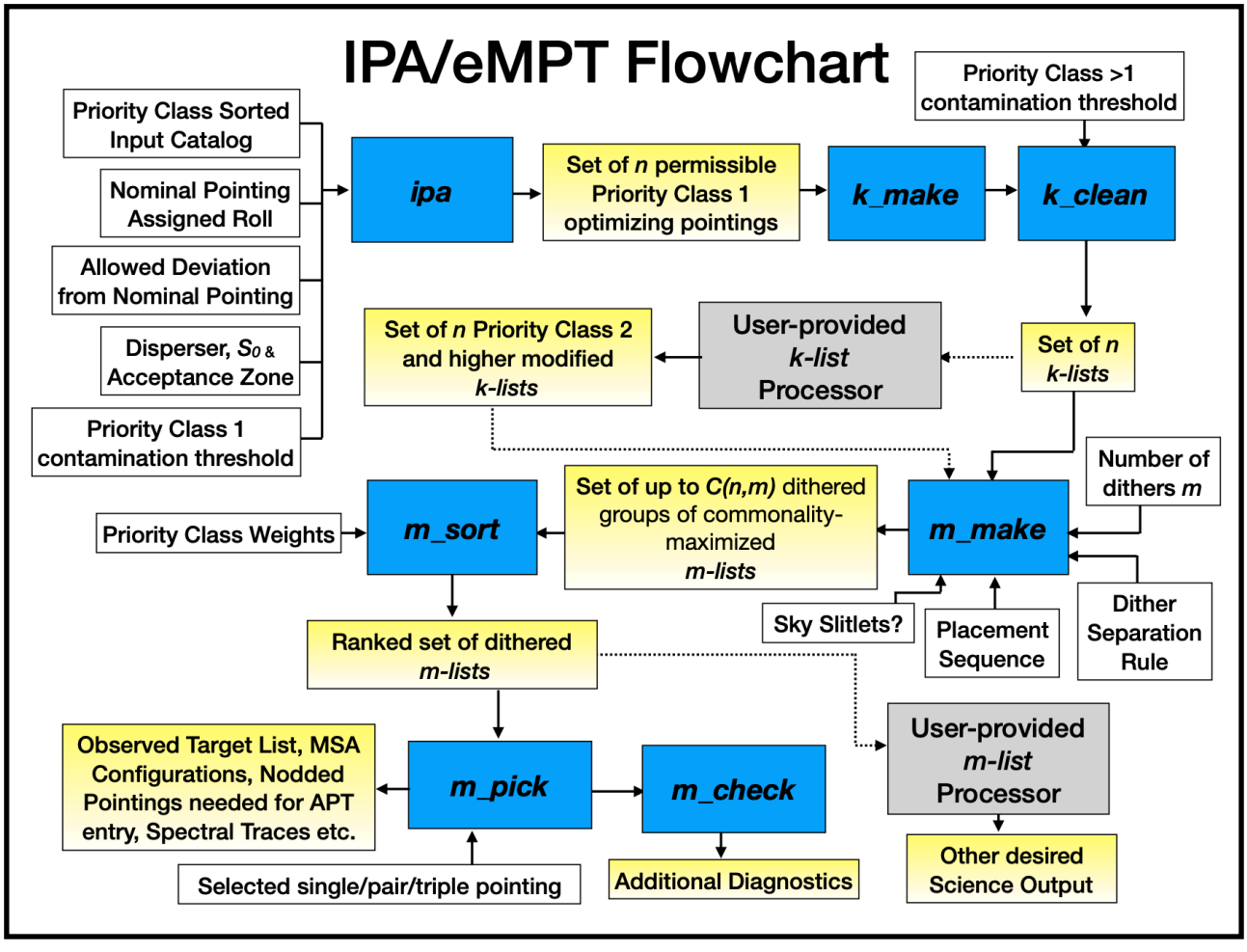
The second and third integers of this parameter value apply to the four nodded shutters that do not nominally contain intended targets (two shutters above and below the central target-containing shutter of the 5-shutter-tall nodded slitlet, respectively):

0: No object allowed in shutter
2: Spill-over from intended target allowed in shutter
4: One non-target object allowed in shutter
8: Two non-target objects allowed in shutter
16: Three non-target objects allowed in shutter

(j) The maximum acceptable Priority Class 2+ contamination scores employing the same contamination scoring scheme as in (i)

1.2 Description of eMPT workflow

The eMPT workflow is dictated by each module in turn, as each appends to the configuration file a short summary of its key output, followed by suggested default settings for the following module. These automatically suggested settings will not necessarily be optimal under all circumstances, and can therefore readily be manually edited and modified by the user as required. The configuration file is in this way intended to serve both as the control interface to the software modules, and as a detailed record of how the resulting MSA mask matching a given NIRSpec MSA observation was produced.



The software suite presently assumes throughout that all targets are to be observed in fully functional three-shutter-tall slitlets in the standard 'nodded' manner.

The seven core executables of the eMPT, that must be run in the sequence shown, perform the following functions:

1) ***ipa*** - Identifies the specific NIRSpec pointings contained within the allowable range of deviation from the nominal pointing that maximize the simultaneous coverage of the Priority Class 1 targets found in the user-supplied input target catalog.

2) ***k_make*** - Identifies all targets of all Priority Classes listed in the input catalog (a) that fall within the specified Acceptance Zone (f) of Viable Slitlets for the observing mode specified in (3.1.3.d) for all the PC1-optimizing pointings that the user specifies in the configuration file. **The so-called *k-list* generated by *k_make* therefore contains the subset of the input catalog targets located in the Acceptance Zone of Viable Slitlets at a given pointing and roll angle.**

3) ***k_clean*** - Identifies and flags all targets in the raw k-list output by *k_make* that turn out to be contaminated above the levels set by the thresholds given in (3.1.3.i) and (3.1.3.j) above at each *ipa*-identified optimal pointing. **It is important to consider that if contamination checking for PC1 targets has been enabled, there is the risk of eliminating the very PC1 targets that were used by the *ipa* module to maximize the optimal pointings that thread through all subsequently run eMPT modules.** If this turns out to be the case, it is at this stage that the user can consider manually setting the priority class of the contaminated PC1 targets from the eMPT input catalog negative (-1) and re-running the *ipa* from scratch until all optimized pointings are based on 'clean' PC1 targets.

4) ***m_make*** - Determines for each candidate pointing the subset of the targets in the contamination-purged k-list output by *k_clean* that can be accommodated on the NIRSpec detector array without incurring spectral overlap (as defined by (3.1.3.d) and (3.1.3.g)). The detailed functioning of the *m_make* module differs depending on the number of dithers specified. **The so-called *m-list* generated by *m_make* therefore contains the subset of the k-list targets whose spectra can be accommodated on the NIRSpec FPA without incurring overlap.**

5) ***m_sort*** - Identifies the optimal single/pair/triple pointing(s) among those output by the *m_make* module in the m-list file; it does this by calculating and assigning a suitable figure-of-merit value to each single/pair/triple pointing and ranking the results accordingly.

6) ***m_pick*** - Parses the information carried through the processing chain for the selected final optimal single/pair/triple pointing identified by the *m_sort* module, and assemble the information required to produce various diagnostic tables and plots summarizing the nature of the chosen solution. Its also generates the information needed to input the resulting MSA mask, detailed pointing(s), and observed target list(s) into the STScI APT/MPT.

7) ***m_check*** - Allows the user to sanity-check every target observed in the pointing(s) identified by the preceding *m_pick* module for contamination.

In addition to automatically appending a summary of its output to the configuration file, each module also keeps the user appraised of its progress by mirroring input parameters and outputting key results and other pertinent information (including any fatal error messages) to the terminal along the way. This information is also captured in further detail in the various output files produced by each module.

In cases where the user needs to re-run the eMPT software with a different parameter setting somewhere along the chain, the relevant parameters need only to be edited at the appropriate place in the configuration file, and the relevant modules run again in sequence starting with the first downstream module affected by the change. There is no need to delete the previously appended blocks in the configuration file nor the output files produced by any modules affected by the changed parameters. The later will be deleted and overwritten automatically, unless renamed by the user.

2 Installing the eMPT

2.1 Software Prerequisites

The software installation instructions shown below were performed on a MacBook Apple M1 Pro machine running the operating system (macOS) Ventura version 13.0 with 32 GB of memory.

2.1.1 Fortran compiler

The eMPT Fortran modules require the *gfortran* Fortran compiler, which is sometimes, but not always, included in a default *gcc* Unix system library (*gcc* is automatically installed on MacOS as part of the Xcode Command Line Developer Tools, but is missing the *gfortran* compiler). **Mac M1 users do not need to separately download *gfortran*, as it will automatically be downloaded as a PGPLOT dependency in step 2.1.2b below.** For other Mac users who require a separate *gfortran* download, it can be accessed at *http://hpc.sourceforge.net* or via Homebrew, as shown below.

```
$ brew install gfortran
```

To check the current *gfortran* installation, issue the following commands in a Unix terminal found at /Applications/Utilities/Terminal:

```
$ gfortran –version
GNU Fortran (Homebrew GCC 13.1.0) 13.1.0
Copyright (C) 2023 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO.
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

$ gcc --version
Apple clang version 14.0.3 (clang-1403.0.22.14.1)
Target: arm64-apple-darwin22.1.0
Thread model: posix
InstalledDir: /Applications/Xcode.app/Contents/Developer/Toolchains/
XcodeDefault.xctoolchain/usr/bin
```

If not already installed, the Mac Command Line Developer Tools must be installed (which also must be reinstalled after a major macOS upgrade) with the following command in Terminal:

```
$ xcode-select --install
```

2.1.2 Mac X11 and Fortran PGPLOT graphics libraries

The Mac XQuartz package must also be installed to enable X11 graphics capabilities required by the plotting functions of the eMPT software, along with the PGPLOT Fortran graphics library.

a) The XQuartz application can be installed from https://www.xquartz.org, or through the package manager Homebrew (which may be installed from: https://brew.sh) as follows:

```
$ brew install xquartz
```

b) Both the Homebrew formula of PGPLOT and its *gfortran* dependency may be installed together using the instructions provided at *https://github.com/kazuakiyama/homebrew-pgplot*, demonstrated below, which is compatible with the Mac M1 system.

```
$ brew install kazuakiyama/pgplot/pgplot
```

Other options include downloading PGPLOT from source according to the instructions at http://mingus.as.arizona.edu/~bjw/software/pgplot_fix.html ; or, via the MacPorts package manager if not accessible through Homebrew. MacPorts is ready to run after downloading the OS-specific installer, which can be found here: https://www.macports.org/install.php. After executing the MacPorts installer, and opening a new Terminal window to receive commands, PGPLOT can be installed and automatically configured (as root user) as shown below (https://ports.macports.org/port/pgplot/),

```
$ sudo port install pgplot
```

After PGPLOT is successfully installed, the PGPLOT_FONT environment variable must be set to the location of the installed *grfont.dat* file in order for text labels to appear on plots it generates:

```
$ export PGPLOT_FONT=`/opt/homebrew/share/grfont.dat' (Homebrew)
                OR
$ export PGPLOT_FONT=`/opt/local/share/pgplot/grfont.dat' (MacPorts)
```

The path to the *libpgplot.a* PGPLOT library must be included in the *gfortran* static linking calls comprising the eMPT Fortran installation script, detailed in section 4.2:

```
$ /opt/local/lib/libpgplot.a (MacPorts)
                OR
$ /opt/homebrew/lib/libpgplot.a (Homebrew)
```

2.2 Compiling the eMPT modules

The full eMPT software suite may be downloaded in a single zip file from its dedicated GitHub page (as *eMPT_v1-main.zip*), consisting of the following files:

- seven Fortran90 source code and associated include files that are compiled into the *ipa*, *k_make*, *k_clean*, *m_make*, *m_sort*, *m_pick*, and *m_check* modules via the associated ASCII compilation shell script *all_compile.sh*;

- eMPT execution shell script named *batch_00.sh* that issues the full sequence of seven newly compiled modules using the template ASCII configuration file named *00.conf*, and writes output to associated subdirectories under directory *trial_00;*

- installation test script *batch_00_truth_test.py* that checks the output of *batch_00.sh* against the expected output from this script stored in the *trial_00_org* directory;

- *reference_files* directory containing additional Fortran90 routines and other references files required by the eMPT, in addition to Python scripts *write_esa_msa_map.py*, *write_msa_metrology.py*, and *read_siaf.py* and their respective ASCII output files; and the Fortran scripts *update_model.sh* and *write_derived_parameters.sh* for updating the instrument model parameters according to the procedure summarized in Section 4.4.1.

The eMPT software is available for download on the ESA Github page:
`https://github.com/esdc-esac-esa-int`

The software is installed in the top-level directory by executing the Fortran compile script *all_compile.sh* in Terminal. The user may need to edit this script directly if being compiled on a hardware and/or software setup that differs form the example used in this installation guide, to ensure that the required PGPLOT, *gcc*, and X11 libraries are being linked from the corresponding local library directories.

```
$ more all_compile.sh

#! /bin/bash
set -e

gfortran -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o ipa
ipa_v2.4.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o k_make
k_make_v1.7.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o k_clean
k_clean_v1.7.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o m_make
m_make_v2.0.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o m_sort
m_sort_v1.7.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o m_pick
m_pick_v2.4.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o m_check
m_check_v1.7.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc   -L/usr/X11/lib -lX11

gfortran  -std=legacy -ffree-line-length-none -fno-backslash -fno-range-check -O2 -o
m_check_regions m_check_regions_v1.6.f90 -L/opt/homebrew/lib -lpgplot -L/opt/homebrew/lib -lgcc
-L/usr/X11/lib -lX11


rm *.mod
```

Note: *Anaconda users may experience a linking error after executing this script if the system 'ld' linker being called is from the Anaconda distribution. In this case, the resolution is to add '/usr/bin' to the start of the path, as shown below, or to deactivate 'conda' before compiling the script.*

```
% which ld
```

/Users/name/opt/anaconda3/bin/ld

```
% export PATH=/usr/bin:$PATH
% which ld
```

/usr/bin/ld

This installation script consists of a set of *gfortran* commands that compile the Fortran90 source code files into executable modules via static links to the location of the *gcc* compiler and PGPLOT, and X11 graphics libraries. It will quietly compile if it is successful, i.e. it should finish without printing any messages or other activity to the screen.

```
$ unzip eMPT_v1-main.zip
$ cd eMPT_v1-main
$ chmod +x all_compile.sh
$ ./all_compile.sh
```

Note: *The following warning message after compilation may safely be ignored; it refers to the fact that, in* gcc *version 10 and onward, it is no longer legal to enter a single scalar to a subroutine that expects an array as input and vice-versa.*

Warning: Rank mismatch between actual argument at (1) and actual argument at (2) (scalar and rank-1)

2.3 Testing the installation

The installation of the eMPT and its prerequisite software may be considered successful and complete when it is confirmed that *batch_00.sh* runs to completion, and its follow-up Python test script *batch_00_truth_test.py* verifies that it exactly reproduced the corresponding verified output in the *trial_00_org* directory.

The *batch_00.sh* script runs the full suite of modules in sequence, using a test set of frozen instrument model parameters, and a single representative use case: all default settings with no dithers. The script *batch_00_truth_test.py* then performs a line-by-line comparison of corresponding key ASCII-formatted module output files in the *trial_00* and *trial_00_org* directories to identify any file mismatches. It also opens the many PS plots generated by the *ipa, m_pick, and m_check* modules when run with the 'y' option (in place of the 'n' choice shown below), so that the user may manually check that they were properly generated.

```
$ more batch_00.sh
#! /bin/bash
./ipa 00.conf
./k_make 00.conf
./k_clean 00.conf
./m_make 00.conf
./m_sort 00.conf
./m_pick 00.conf
./m_check 00.conf
```

```
$ ./batch_00.sh

[pages of screen output by the eMPT modules]
.
.
.
```

```
$ python batch_00_truth_test.py n

Comparing ./trial_00_org/ipa_output/grouped_optimal_pointings.txt and ./trial_00/
ipa_output/grouped_optimal_pointings.txt for T or F match:
True
Comparing ./trial_00_org/k_clean_output/k_list_mod.txt and ./trial_00/k_clean_output/
k_list_mod.txt for T or F match:
True
Comparing ./trial_00_org/k_make_output/k_list_raw.txt and ./trial_00/k_make_output/
k_list_raw.txt for T or F match:
True
Comparing ./trial_00_org/m_make_output/single_m_list.txt and ./trial_00/m_make_output/
single_m_list.txt for T or F match:
True
Comparing ./trial_00_org/m_sort_output/single_list_fom2.txt and ./trial_00/
m_sort_output/single_list_fom2.txt for T or F match:
True
Comparing ./trial_00_org/m_pick_output/pointing_100//observed_targets.cat and ./
trial_00/m_pick_output/pointing_29//observed_targets.cat for T or F match:
True
Comparing ./trial_00_org/m_pick_output/pointing_100//pointing_summary.txt and ./
trial_00/m_pick_output/pointing_29//pointing_summary.txt for T or F match:
True
Comparing ./trial_00_org/m_pick_output/pointing_100//shutter_mask.csv and ./trial_00/
m_pick_output/pointing_29//shutter_mask.csv for T or F match:
True
Comparing ./trial_00_org/m_pick_output/pointing_100//slitlet_usage_stats.txt and ./
trial_00/m_pick_output/pointing_29//slitlet_usage_stats.txt for T or F match:
True
Comparing ./trial_00_org/m_pick_output/pointing_100//target_list.txt and ./trial_00/
m_pick_output/pointing_29//target_list.txt for T or F match:
True
```

3. Running the eMPT

3.1 Executing the Fortran modules

The core Fortran modules of the eMPT are executed on the Unix command line in the exact manner and strict sequence shown below, specifying the name of the controlling configuration file after naming each. In this example, the trial run has been identified as trial “01” (in the user-specified parameter of the configuration file, item 3.1.3.c above) and therefore the name of the configuration file is named to match.

```
$ ./ipa 01.conf
$ ./k_make 01.conf
$ ./k_clean 01.conf
$ ./m_make 01.conf
$ ./m_sort 01.conf
$ ./m_pick 01.conf
$ ./m_check 01.conf
```

After each module is run, it writes out various module settings and output summary information

to the screen and the configuration file, in sequence, and summary and diagnostic ASCII and PS plot files to the trial run module sub-directories, as presented in Section 4.5 and the Appendix.

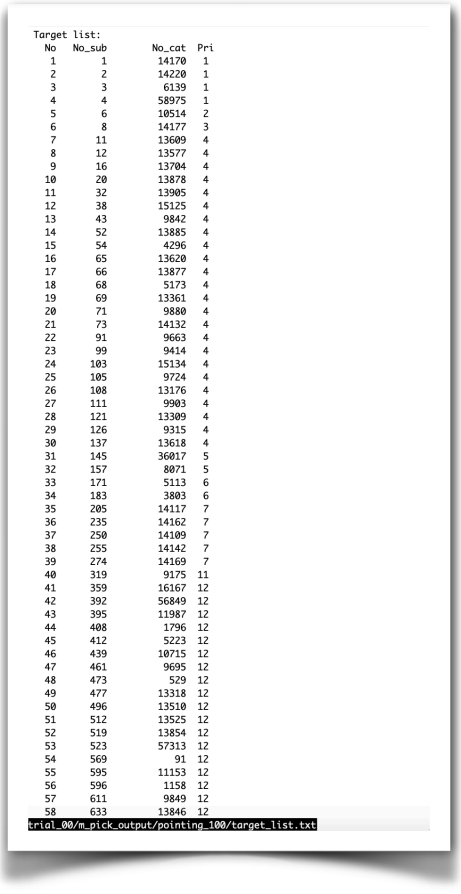
3.3 Examining the software output

3.3.1 Primary output

After a completed trial of the eMPT software, the user can examine the configuration file for a concise record of module settings and activity describing the run, and consider all output in detail by inspecting the summary and diagnostic plot files written to the trial run output directories by each module (see the Appendix for an exhaustive list with associated descriptions of all module output files). The primary output of the eMPT is the single pointing, or pair or triplet of dithered pointings that optimize(s) the observation at hand, and the associated MSA slit configuration and final target list summary information needed to enter the observation into the STScI APT/MPT system.

The key output products delivered by the *m_pick* module include:

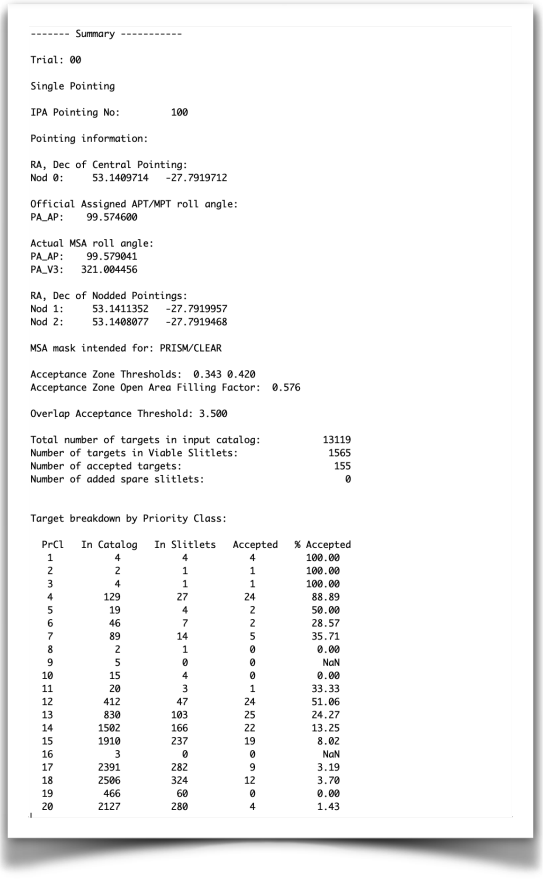
- Catalog of observed targets
- Pointing summary with target slitlet assignments
- APT-formatted MSA CSV configuration file
- Map of final MSA configuration
- Plot of observed targets and MSA outline projected to the sky
- Spectral traces for all seven NIRSpec dispersers



Target list:

No	No_sub	No_cat	Pri
1	1	14170	1
2	2	14220	1
3	3	6139	1
4	4	58975	1
5	6	10514	2
6	8	14177	3
7	11	13609	4
8	12	13577	4
9	16	13704	4
10	20	13878	4
11	32	13905	4
12	38	15125	4
13	43	9842	4
14	52	13885	4
15	54	4296	4
16	65	13620	4
17	66	13877	4
18	68	5173	4
19	69	13361	4
20	71	9880	4
21	73	14132	4
22	91	9663	4
23	99	9414	4
24	103	15134	4
25	105	9724	4
26	108	13176	4
27	111	9903	4
28	121	13309	4
29	126	9315	4
30	137	13618	4
31	145	36017	5
32	157	8071	5
33	171	5113	6
34	183	3803	6
35	205	14117	7
36	235	14162	7
37	250	14109	7
38	255	14142	7
39	274	14169	7
40	319	9175	11
41	359	16167	12
42	392	56849	12
43	395	11987	12
44	408	1796	12
45	412	5223	12
46	439	10715	12
47	461	9695	12
48	473	529	12
49	477	13318	12
50	496	13510	12
51	512	13525	12
52	519	13854	12
53	523	57313	12
54	569	91	12
55	595	11153	12
56	596	1158	12
57	611	9849	12
58	633	13846	12

trial_00/m_pick_output/pointing_100/target_list.txt



------- Summary ----------

Trial: 00

Single Pointing

IPA Pointing No: 100

Pointing information:

RA, Dec of Central Pointing:
Nod 0: 53.1409714 -27.7919712

Official Assigned APT/MPT roll angle:
PA_AP: 99.574600

Actual MSA roll angle:
PA_AP: 99.579041
PA_V3: 321.004456

RA, Dec of Nodded Pointings:
Nod 1: 53.1411352 -27.7919957
Nod 2: 53.1408077 -27.7919468

MSA mask intended for: PRISM/CLEAR

Acceptance Zone Thresholds: 0.343 0.420
Acceptance Zone Open Area Filling Factor: 0.576

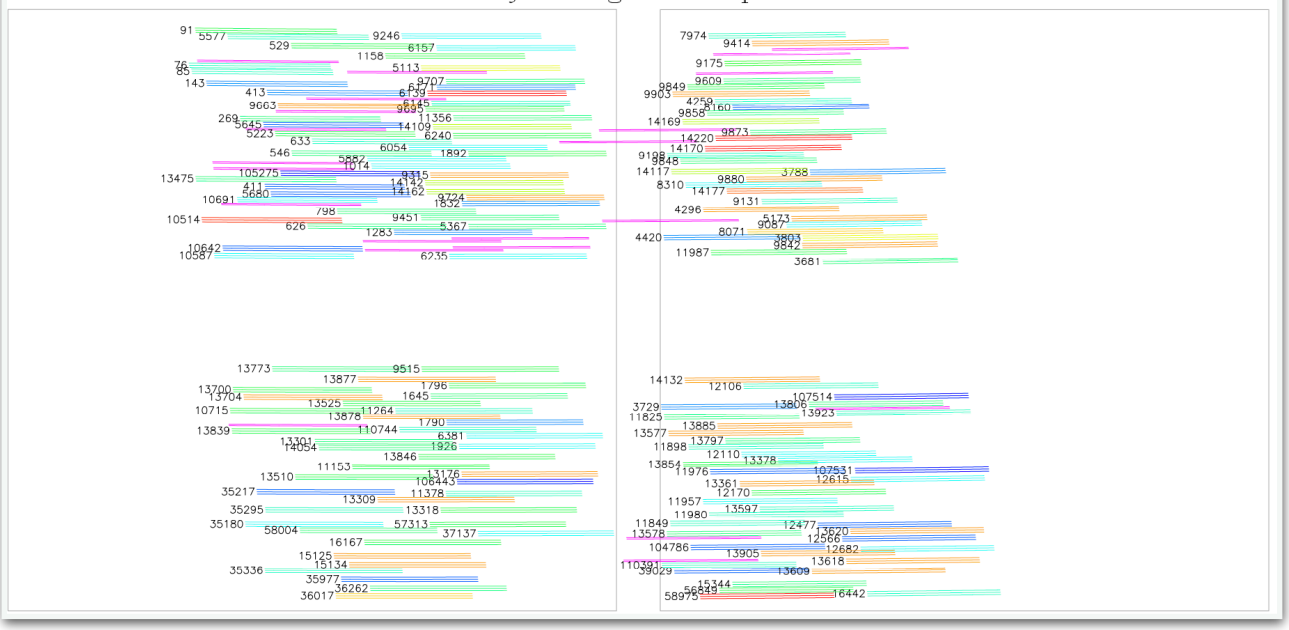
Overlap Acceptance Threshold: 3.500

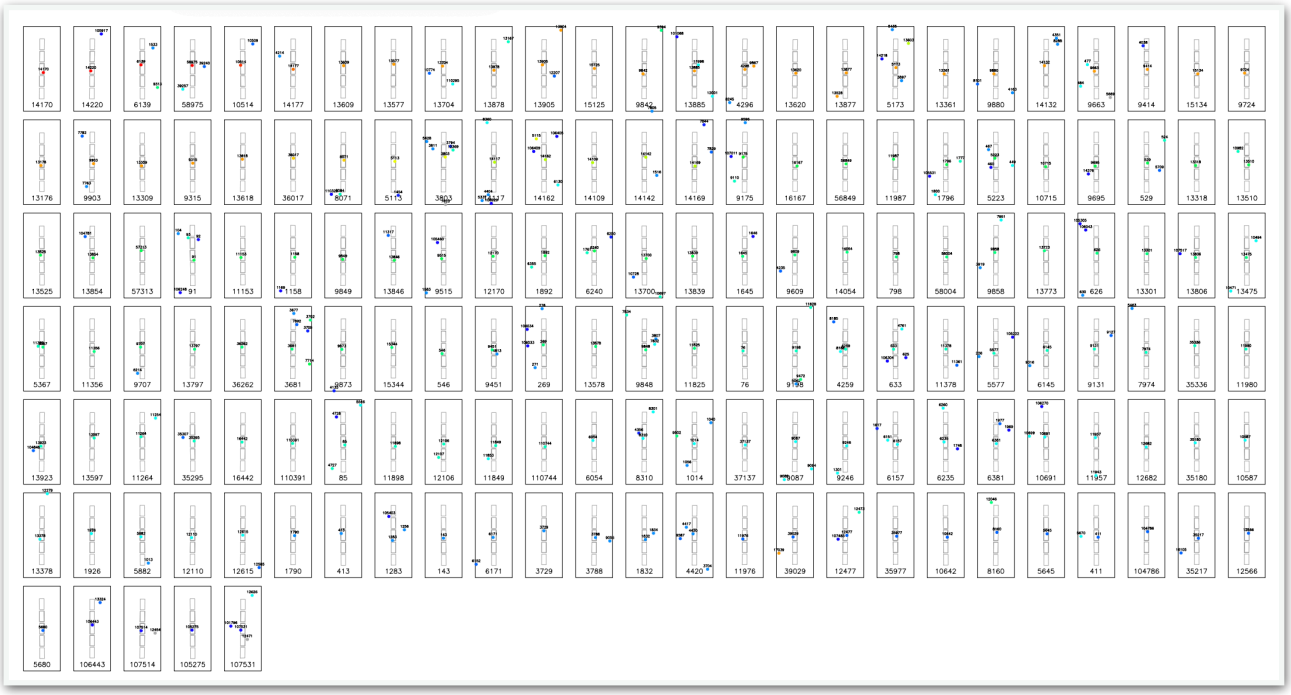
Total number of targets in input catalog: 13119
Number of targets in Viable Slitlets: 1565
Number of accepted targets: 155
Number of added spare slitlets: 0

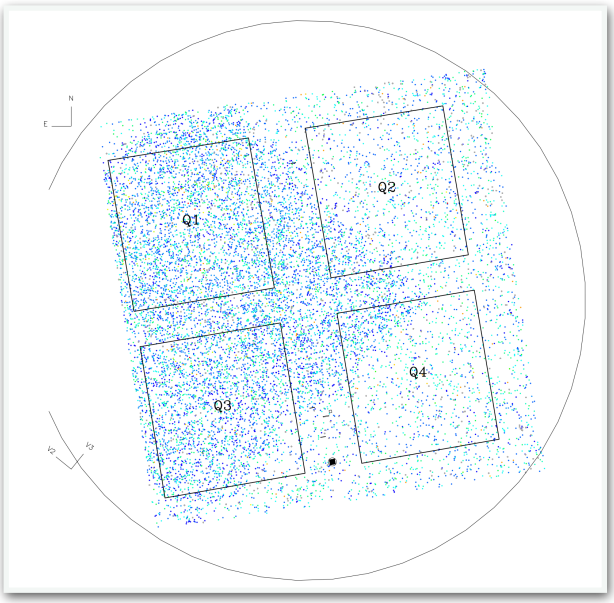
Target breakdown by Priority Class:

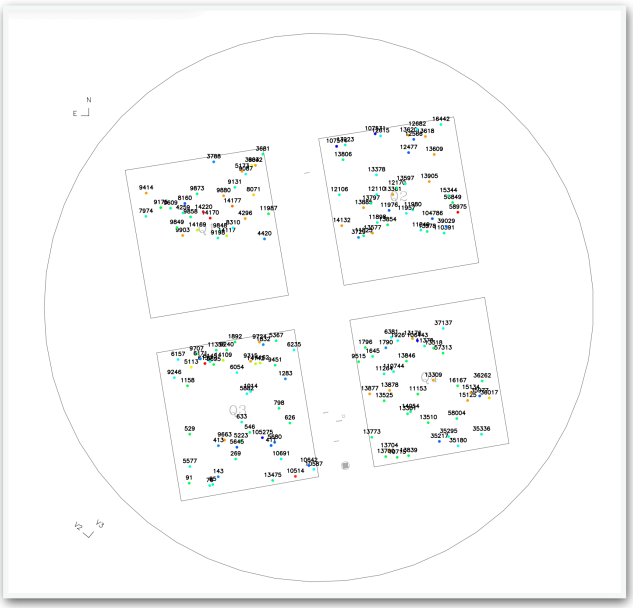
PrCl	In Catalog	In Slitlets	Accepted	% Accepted
1	4	4	4	100.00
2	2	1	1	100.00
3	4	1	1	100.00
4	129	27	24	88.89
5	19	4	2	50.00
6	46	7	2	28.57
7	89	14	5	35.71
8	2	1	0	0.00
9	5	0	0	NaN
10	15	4	0	0.00
11	20	3	1	33.33
12	412	47	24	51.06
13	830	103	25	24.27
14	1502	166	22	13.25
15	1910	237	19	8.02
16	3	0	0	NaN
17	2391	282	9	3.19
18	2506	324	12	3.70
19	466	60	0	0.00
20	2127	280	4	1.43

PRISM 155 Target Spectra
0 Sky Background Spectra









3.3.2 Configuration file summary and user edit options

Before an initial run of the eMPT software, its controlling configuration file appears in its starting template form, with only the user-input and pre-set initial input parameters in place. Comment lines are preceded by '#' and the user is free to add comments or notes to the configuration file as needed.

```
# This is the template IPA/eMPT configuration file
#
# Rules: Lines starting with # are ignored
#        No empty or blank lines
#        Multiple input parameters in line must be separated by one or more spaces
#        The exxact sequence of input parameters must be adhered to
#        The data type of each entry (INTEGER, REAL os STRING) must be adhered to
#
#
# Unique ID number of the trial [integer, 00:99]
   00
# The output files generated by each module go into separate subdirectories within the topmost
created directory ./trial
#
# NIRSpec disperser employed
# Specifies the disperser driving the MSA mask design such that no overlap of spectra occur
# legal entries: [string:
#   PRISM/CLEAR
#   G140M/F070LP ,G140M/F110LP, G235M/F170LP, G395MF290LP
#   G140H/F070LP ,G140H/F110LP, G235H/F170LP, G395HF290LP ]
  PRISM/CLEAR
#
# Acceptance Zone half extents in MSA X and Y in units of MSA facets (reals)
# Full shutter range: [0.5 0.5]
# Shaved full open area (default): [0.343 0.420]
  0.343   0.420
#
# Minimum vertical spectral separation threshold in MSA shutter Y facets [real, Default: 3.5]
  3.5
#
# Name of eMPT-formatted target input catalog [string up to 70 characters long, including relative
path if needed]
 test_trial_00.cat
#
# Name of astrometrically calibrated fits reference image covering the entire MSA FOV to
# be used for advanced k_clean target sorting and the rendering of "Rogue's Gallery"
# plots. It is imperative that the reference image have an STScI-style WCS solution in its
# header on the same absolute celestial reference system as that of the target coordinates
# in the input catalog.
# [string up to 100 characters long, including relative path if needed]
# Enter "none" when not available or if not needed:
  none
#
# RA, Dec & PA_AP of nominal pointing in decimal degrees [reals, RA 0:360 - Dec -90:90 - PA_AP
0:360]
# NB! The value of PA_AP roll angle to be entered is the official STScI ASSIGNED value -
# i.e. NOT the actual MSA roll angle.
53.141047531  -27.791982467   99.5746
#
#
# Allowable absolute deviation from nominal pointing in arcsec in X and Y on MSA [reals. Defaults
+/-25.0 x +/-25.0]
   10.   10.
#
# Angle between JWST spacecraft velocity vector and the nominal NIRSpec pointing in
# decimal degrees to be needed to calculate the change in MSA plate scale caused by
# Differential Velocity Aberration.
# This angle depends on the the epoch of the observation implicitly given by the roll
# angle PA_AP assigned to the observation. It is calculated by the STScI MPT where
# it appears  as "True angle to target".
# [real, decimal degrees, -180, +180]:
  80.7829
#
# Contaminated target elimination score thresholds
#
# Shutters nominally containing intended targets (first number):
#
```

```
#  1: Intended target in shutter
#  4: One non-target object present in shutter
#  8: Two non-target objects present in shutter
# 16: Three non-target objects present in shutter
#
# Four nodded shutters not nominally containing intended targets (2nd and 3rd numbers):
#
#  0: No object present in shutter
#  2: Spill-over from intended target present in shutter
#  4: One non-target object present in shutter
#  8: Two non-target objects present in shutter
# 16: Three non-target objects present in shutter
#
# Maximum acceptable contamination scores for Priority Class 1 targets [integer, Default 1 2 2, set
to 100 100 100 to di
# [integer, Default 1 2 2, set to 100 100 100 to turn off contamination elimination]
# NB! Does not apply to IPA module, so turning on subsequent contamination elimination of
# Priority Class 1 targets can lead to illogical situations where the IPA pointings to be further
# are optimized for targets that have since been eliminated. Should therefore normally be
# kept turned off.
  100 100 100
#
# Maximum acceptable contamination scores for Priority Class 2 and larger targets [integer, Default
1 2 2, set to 100 10
# [integer, Default 1 2 2, set to 100 100 100 to turn off contamination elimination]
   1 2 2
#
# Number of dithered pointings sought in above search box [integer, 1, 2 or 3]
   1
#
#  -- Modifying any parameter in the above section requires re-running the ipa and subsequent
modules ---
```

When a run of the eMPT software has been completed, the configuration file appears updated with output summary information from each of the modules in the order in which they were executed. **It is at this point that the user can decide that they are satisfied with the results of the initial run of the software and can proceed to the next step of entering the *m_pick* output into the STScI APT/MPT system; or, change parameter settings for one or more of the modules as described below and re-run the software as often as needed until the optimal scientific results are obtained.**

The information that is appended to the configuration file by each module is described below, with associated snapshots of an example configuration file that was run for an observation requiring three dithers. **Places in the configuration file where the user has the option to edit the module output midstream in the eMPT workflow and re-run the software through to that point in order to alter the course of the full run, are noted and user options provided.**

ipa output

When the ipa module has finished running and before the next module in the module chain (k_make) executes, the list of *ipa*-identified PC1-optimizing pointings is automatically appended to the configuration file. Each line in this list gives the (integer) *ipa*-assigned pointing designation, the (integer) simultaneous PC1 target coverage number achieved at the pointing, and the double precision (real) Right Ascension, Declination and (real) roll angle (pa_ap) in decimal degrees.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE IPA MODULE ----------
#
# Total number of observable Priority Class 1 targets in range: 4
#
# Maximum number of Priority Class 1 targets observable simultaneously in a single pointing: 4
#
#
```

```
# A detailed sorted list and map of all optimal pointings identified by the IPA module can be found
in the directory:
#
#   ./trial_00/ipa_output/
#
#
#   1 groups covering 4 targets at 3 or more pointings were found
#
# List of prime candidate optimal pointings automatically generated by the IPA module:
#     (selected as all pointings achieving the highest possible simultaneos priority
#     class 1 target coverage and belonging and groups of pointings covering the same
#     targets at n_dither or more pointings)
#
#  Consult ./trial_00/ipa_output/grouped_optimal_poinings.txt for lists of
#      which priority 1 targets are covered by the individual pointings of each group
#
#IL# -- Marker. Do not delete or move this linoo0-
#
# Pointing
#  No. Coverage   RA [deg]       Dec [deg]     pa_ap [deg]   Group No.
     1     4     53.140153926  -27.789124075   99.579406738       1
     2     4     53.140319805  -27.789148846   99.579345703       1
     3     4     53.141480559  -27.789322184   99.578796387       1
     4     4     53.140017567  -27.789221258   99.579467773       1
     5     4     53.140181435  -27.789244957   99.579406738       1
     6     4     53.140345287  -27.789268679   99.579345703       1
     7     4     53.141459887  -27.789429925   99.578796387       1
     8     4     53.141445752  -27.789503591   99.578796387       1
     9     4     53.140117958  -27.789311528   99.579467773       1
    10     4     53.140283983  -27.789335543   99.579345703       1
.
.
.
   233     4     53.137922901  -27.791344980   99.580444336       1
   234     4     53.137894645  -27.791491598   99.580505371       1
   235     4     53.137866391  -27.791638218   99.580505371       1
#
#
#IE# -- Marker. Do not delete or move this line
#
#
# CAUTION: The above automatically selected candidate pointing list is not necessarily optimal
depending on the task at hand
#
#             Consult the complete IPA output in ./trial_00/ipa_output/
#
# ----- END OF SECTION APPENDED BY THE IPA MODULE ------
```

Depending on the objectives of the user, it will often be advantageous to edit this automatically generated list by hand, and instead construct a more informed (and shortened) set of pertinent candidate optimal pointings through consulting the pointing map (./trial_num/ipa_output/map_optimal_pointings.ps) and the grouped listing (./trial_num/ipa_output/grouped_optimal_poinings.txt) above with an eye to the particulars of the NIRSpec observation in question.

k_make output

When the *k_make* module finishes running, it appends to the configuration file the path to the raw k_list file that lists the (integer) target ids, Priority Classes, MSA slitlet designations and (real) relative intra-shutter target locations for all targets located within Viable Slitlets (as defined by the observing mode and set Acceptance Zone for each pointing in the list.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE K_MAKE MODULE ----------
#
# The generated k-list file is located in:
#
#    ./trial_01/k_make_output/k_list_raw.txt
#
```

k_clean output

The *k_clean* module automatically appends to the configuration file the prioritized order in which the spectra of the targets are to be placed on the detector, in each of the natural groupings making up the sorted target k-list set at each pointing under consideration. The default placement sequence assumes that the user wishes to maximize target commonality between all pairs of dithered pointings (and thereby the average exposure time of the observed targets) by first placing all the common targets in sequence of increasing Priority Class, followed by the unique targets in the same sequence. *Should the user instead wish to maximize the total number of different targets covered by the pair of pointings, then the two rows should be swapped in the placement sequence*.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE K_CLEAN MODULE ----------
#
# The modified k-list file with contaminated targets marked with negative Priority Class
designations is located in:
#
#    ./trial_00/k_clean_output/k_list_mod.txt
#
#
#
# The following suggested default parameters should be reviewed and modified as needed before
running the m_make module:
#
# Number of Priority Classes present in Input Catalog:
          20
#
#
# Order in which segmented k-lists are to be fed to the Arribas algorithm
# Default sequence in strict order of Priority Class:
           1           2           3           4           5           6           7           8
9          10          11          12          13          14          15          16          17
18          19          20
#
#
#
# ----- END OF SECTION APPENDED BY THE K_CLEAN MODULE ------
```

The *k_clean* module also appends to the configuration file the default set of dither censoring thresholds. These values may be tightened or cautiously loosened by the user if needed in order to ensure that the number of legal dithers is kept greater or equal to the number of dithers, bearing in mind the the code will work through arbitrarily large numbers of legal single, pair or triple pointings and slow down considerably. Alternatively, the list of *ipa*-identified PC1-optimizing pointings can be revisited and reduced or increased in number as needed.

m_make output

The *m_make* module automatically appends to the configuration file the path to the m-list file listing the accepted targets and matching slitlets for each processed pair or triple pointing, along with the suggested row of Priority Class weights to be used for calculating the figure-of-merit that will be used to rank and sort the optimal single/pair/triplet pointings. The default setting represents a "top heavy" weighting scheme in which one Priority Class 1 object is "worth" two Priority Class 2 objects, 4 Priority Class 3 objects, 8 Priority Class 4 objects, and so forth. *The user is entirely free to edit these weights to better reflect the scientific objectives of the program in question, keeping in mind the placement sequence specified above*.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE M_MAKE MODULE ----------
#
# The generated m-list file is located in:
#
#   ./trial_00/m_make_output/single_m_list.txt
#
# Number of pointings:        235
#
# The following default parameters should be reviewed and modified as needed before running the
m_sort module:
#
# Figure of Merit Weights for each Priority Class in increasing order
   1.00000000      0.500000000      0.250000000      0.125000000       6.25000000E-02
3.12500000E-02   1.56250000E-02   7.81250000E-03   3.90625000E-03   1.95312500E-03   9.76562500E-04
4.88281250E-04   2.44140625E-04   1.22070312E-04   6.10351562E-05   3.05175781E-05   1.52587891E-05
7.62939453E-06   3.81469727E-06   1.90734863E-06
#
# ----- END OF SECTION APPENDED BY THE M_MAKE MODULE ----------
#
#  -- Modifying any parameter in the above section requires re-running the m_sort and following
modules -
```

m_sort output

The *m_sort* module automatically appends to the configuration file the paths to the figure-of-merit ranked single/pair/triple pointing lists for each of “FOM1” (weighted average exposure time per observed target) and “FOM2” (weighted total number of targets observed), printing the FOM1 ranked list explicitly with the top entry "uncommented" in the list.

In the case of an observation with no dithers, the default suggested chosen pointing is the one maximizing FOM2. In the case of two or three dithers, the default suggested dithered pointing pair or pointing triple is the one maximizing FOM1 and thereby commonality of targets between the dithered pointings.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE M_SORT MODULE ----------
#
# The figure-of-merit ranked single pointing list is located in:
#
#   ./trial_00/m_sort_output/single_list_fom2.txt
#
#
# The following default parameters should be reviewed and modified as needed before running the
m_pick module:
#
# Top ranked (FOM2) pointings:
#         Pointing   FOM2
    100    100     8.03851
#    116    116     7.83266
#    106    106     7.69332
#    111    111     7.67813
#    123    123     7.64708
#     83     83     7.59984
#    122    122     7.58675
#     87     87     7.55332
#    174    174     7.54103
#    103    103     7.53797
#
# Automatically fill out final MSA mask with Sky Background Slitlets?
# [Y or N [Default] ]
  N
#
#
# ----- END OF SECTION APPENDED BY THE M_SORT MODULE ------
```

It should be stressed, however, that the user is entirely free to select a different preferred optimal solution than the proposed one by "commenting" the proposed solution by adding a hash at the start of the line and "uncommenting" a different line in the configuration file - or selecting an altogether different preferred solution by copying and pasting the relevant entry in the complete ranked FOM listings produced by the m_sort module.

Finally, the output of the *m_sort* module to the configuration file offers the user the option of utilizing any un-used (and uncontaminated) Viable Slitlets remaining on the MSA after target placement to place as many non-overlapping "empty" slitlets as it can, for the purpose of improving the measure of the spectrum of the sky background. *The default option automatically augmented to the configuration file is "N", but this can manually be changed to "Y" if preferred.*

m_pick output

The *m_pick* module automatically appends to the configuration file the pointing coordinates and associated target coverage numbers for the singe/pair/triple pointings designated by the previous *m_sort* module as the final optimal ones(s). If the user chose in the previous step to utilize unused and uncontaminated Viable Slitlets remaining on the MSA after target placement, the number of Sky Background slitlets added to the masks of the final selected pointing(s) is also printed.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE M_PICK MODULE ----------
#
#
# Selected single optimal pointing:
#
# IPA Pointing No:       100
# Coordinates and actual MSA Roll:    53.1409714   -27.7919712    99.579041
# Number of targets:       155
#
#
# Target breakdown by Priority Class:
#
#    Pri   Number
#     1      4
#     2      1
#     3      1
#     4     24
#     5      2
#     6      2
#     7      5
#     8      0
#     9      0
#    10      0
#    11      1
#    12     24
#    13     25
#    14     22
#    15     19
#    16      0
#    17      9
#    18     12
#    19      0
#    20      4
#
# 0 Sky Background slitlets added to the IPA Pointing 100 MSA Mask
#
#
# The full output from the m_pick module can be found in the directory:
#
#    ./trial_00/m_pick_output/pointing_100/
#
#  and in the file:
#
#    ./trial_00/m_pick_output/pointing_100/target_list.txt
#
# ----- END OF SECTION APPENDED BY THE M_PICK MODULE ——————
```

m_check output

The purpose of the m_check module is to allow the user to sanity-check every target observed in the pointing(s) identified by the preceding *m_pick* module for contamination, and therefore appends pointers to the module output file locations for this examination.

```
# ----- SECTION AUTOMATICALLY APPENDED BY THE M_CHECK MODULE ----------
#
#
#
# Refer to the close-up slitlet proximity maps in:
#
#   ./trial_00/m_check_output/pointing_100/slitlet_panel_plot.ps
#
# to gauge the contamination levels of each target at each final pointing.
#
# Targets deemed to be excessively contaminated may be excluded from observation by
# changing the sign of their Priority Class assignments to be negative in the
# input catalog, and running the ipa and subsequent modules anew.
#
#
#
# ----- END OF SECTION APPENDED BY THE M_CHECK MODULE ------
```

4 Ingesting eMPT output into the APT

After opening the APT and loading/initiating an APT file for a given NIRSpec MOS program, the following steps will create an MPT Plan that utilizes an eMPT MSA configuration.

There are two closely related approaches to generating an APT MOS observation with the MPT Planner[1] that utilizes a customized eMPT-generated pointing configuration(s): one involves completely replacing the MSA configuration of an existing MPT Plan with that output by the eMPT; the other results in the 'MPT version' of the custom eMPT mask that may not exactly match the latter.

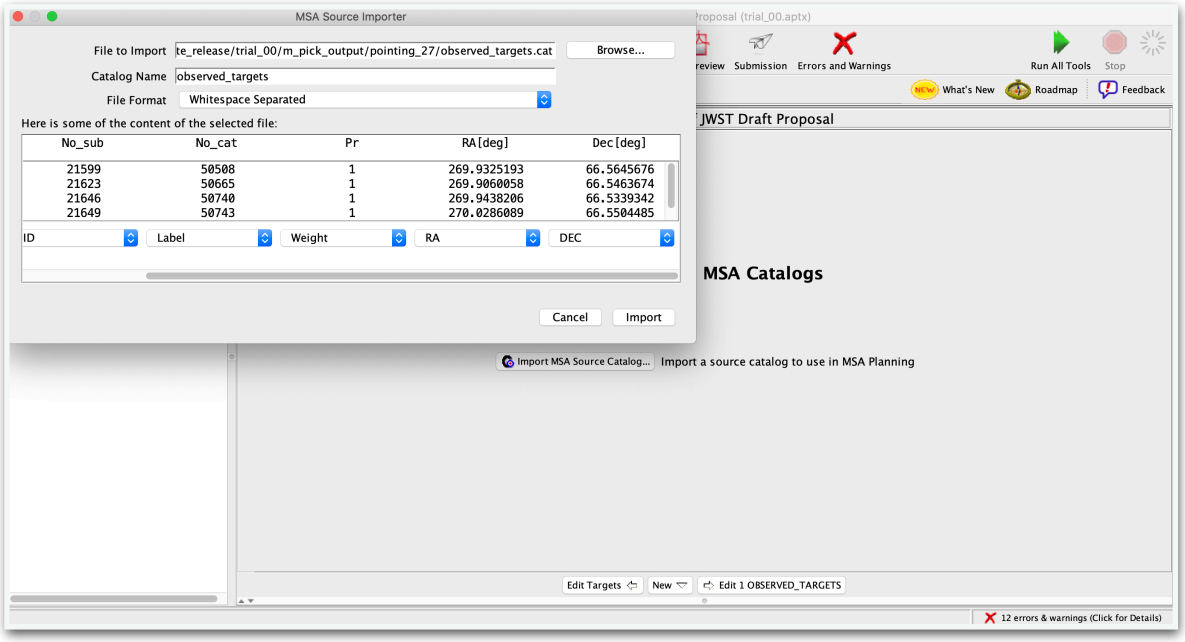
Any detected discrepancies between an eMPT MSA configuration and the corresponding MPT version of that configuration are likely due to the fact that the eMPT does not by default include a correction for differential velocity aberration like the MPT (which can be remedied by re-running the eMPT using the same 'angle to target' the MPT assumes, which can be located in the XML file output of the *File -> Export-> xml* APT function).

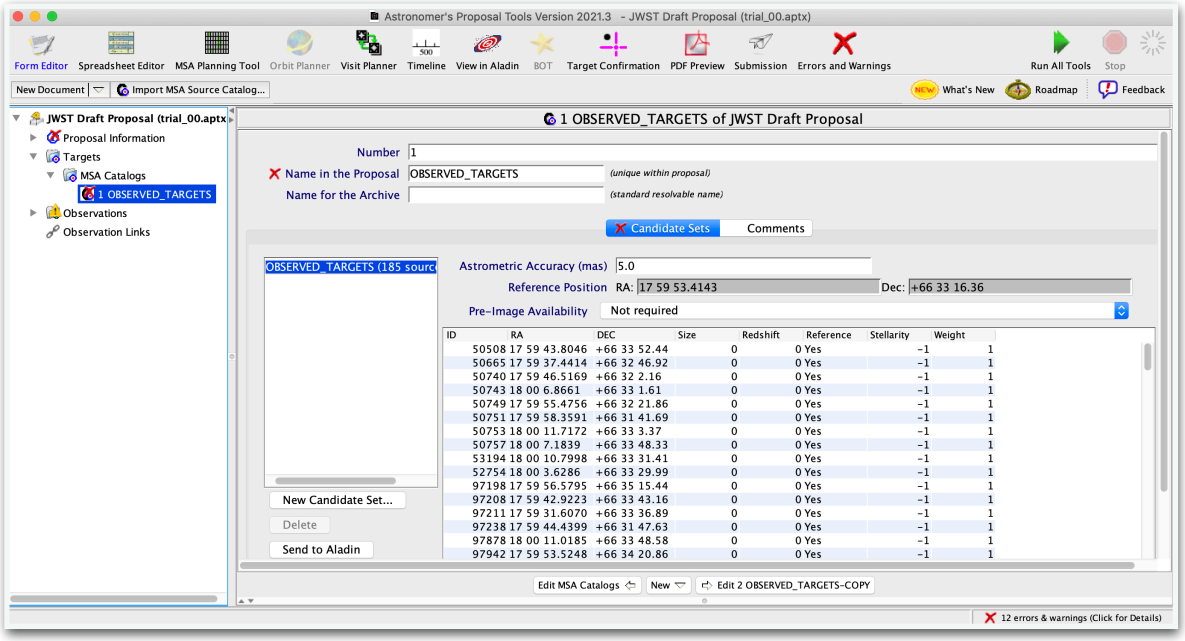
Furthermore, in PRISM mode, the MPT will only select a subset of the targets included in the eMPT-generated configuration due to a different handling of the horizontal overlap of PRISM spectra compared to the eMPT; neither will the MPT mask include any empty sky background shutters included by the eMPT *m_make* module.

---

[1] It is possible in the APT to completely bypass the MPT Planner in creating a custom MOS observation from scratch, for expert users. See detailed instructions here in JDocs: https://jwst-docs.stsci.edu/jwst-near-infrared-spectrograph/nirspec-apt-templates/nirspec-multi-object-spectroscopy-apt-template/custom-mos-observations-using-the-msa-configuration-editor#CustomMOSObservationsusingtheMSAConfigurationEditor-custom-config)

The user is assumed to be familiar with the basic workings of the APT, so the following instructions are intentionally left terse. These instructions are valid for APT Version 2021.3+.

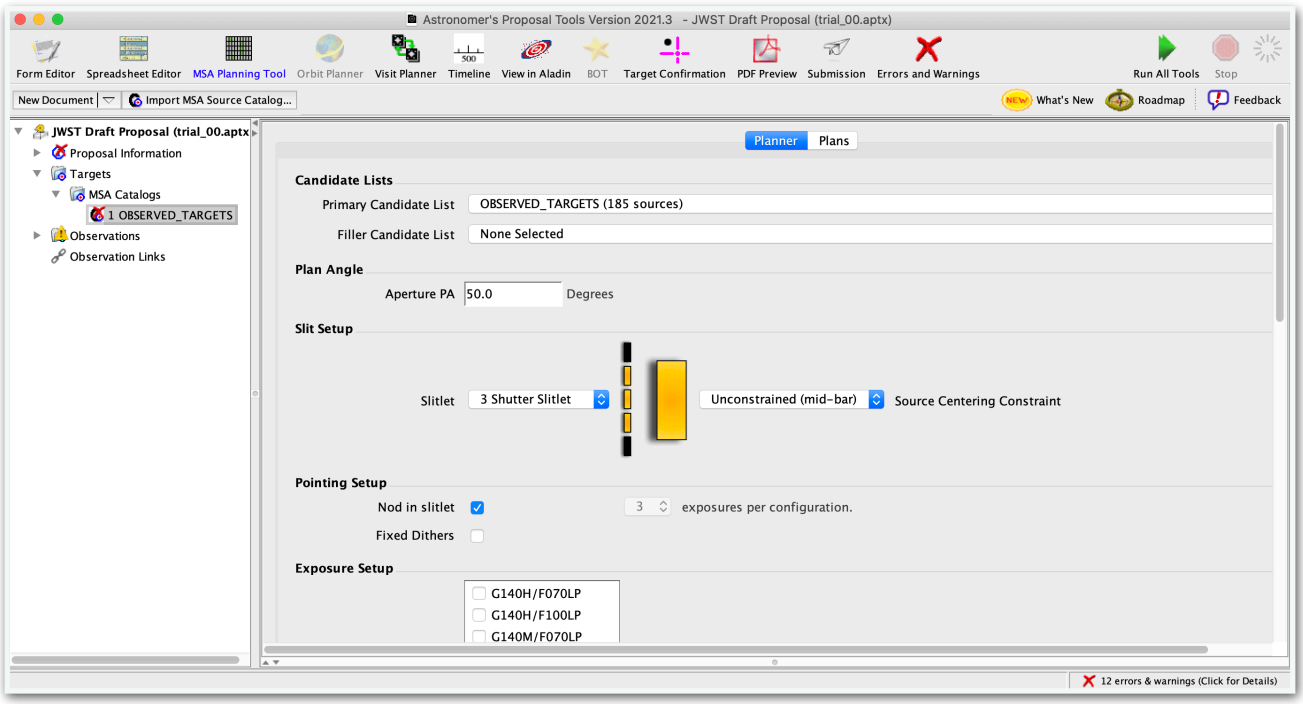




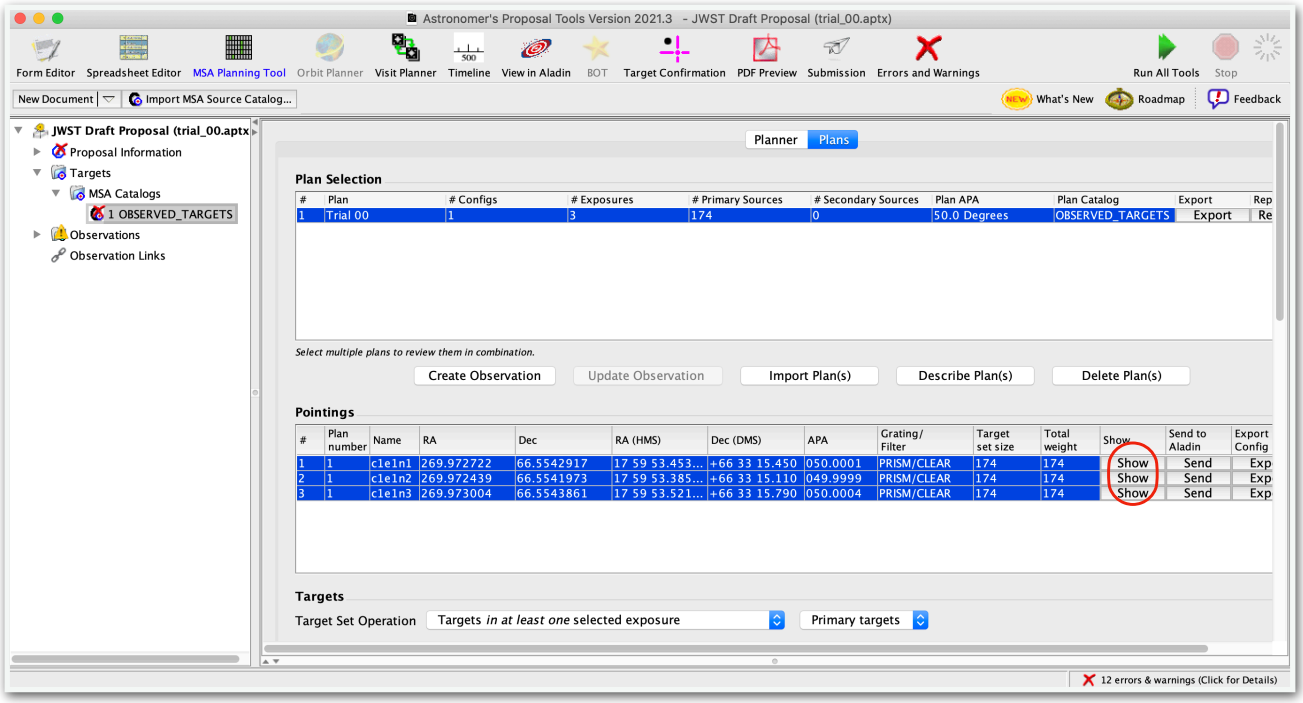
Step 1: Enter the eMPT final observed target list, *observed_targets.cat*, into the MSA Planning Tool via the *Targets -> Import MSA Source Catalog* field in the Form Editor of the APT, selecting Whitespace Separated formatting, and setting the ID, RA, and Dec columns correctly.

After importing the catalog, set the Astrometric Accuracy and Pre-Image Availability fields to avoid the error flags.

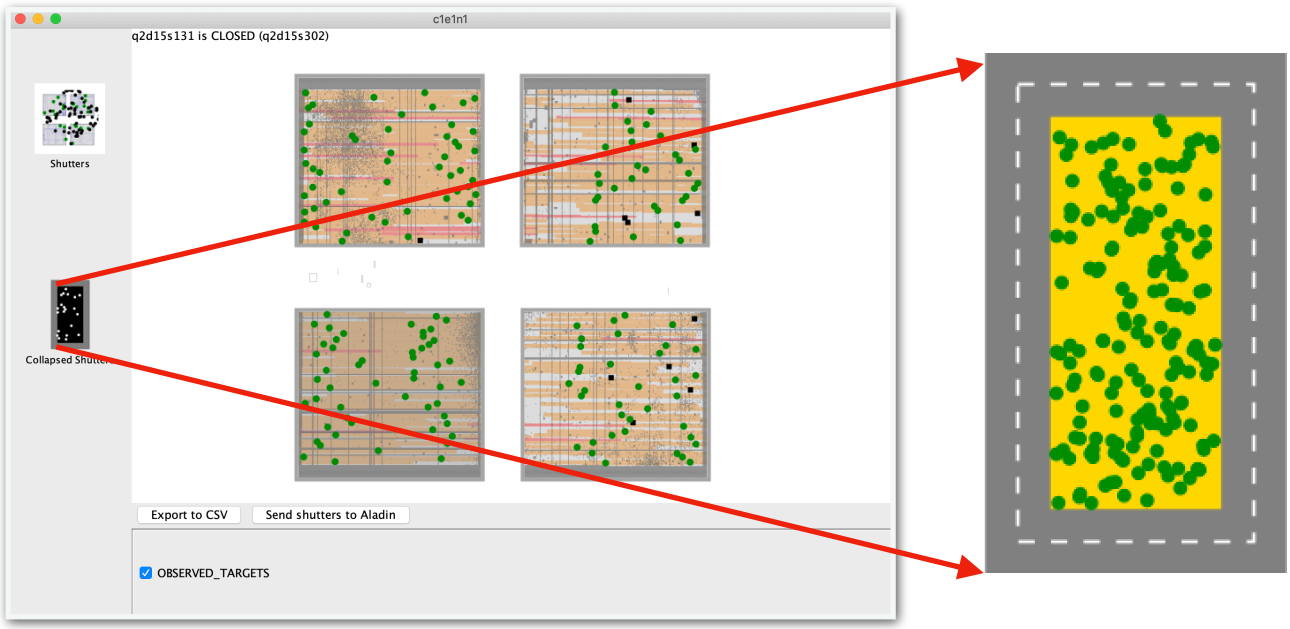
Step 2: In the Planner tab of the MSA Planning Tool, fill in the PA_AP roll angle cut and pasted from *pointing_summary.txt*, select the above target list, select a 3-shutter slitlet and an unconstrained (full facet) Acceptance Zone, Nod in slitlet, No Dither, and complete the Exposure Setup panel to match the current observation. Then in the Search Grid panel, cut and paste the central pointing RA & Dec from *pointing_summary.txt* and disable the search grid by setting the Width and Height fields of the Search Box to 0.0 and setting the Number of configurations to 1. Finally, provide a name for the new plan and press Generate Plan.



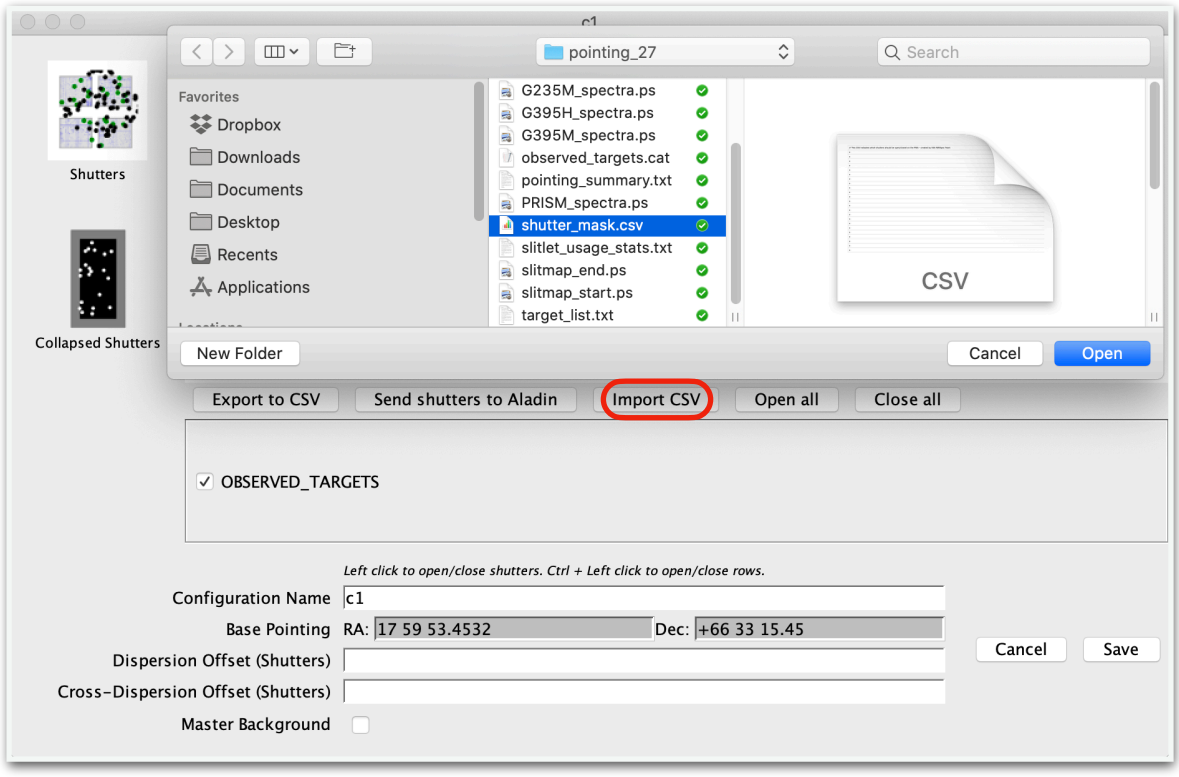
Switching over to the Plans tab of MSA Planning Tool, select Create Observation, and view the MSA configuration using the Show button in each of the rows of the the MSA Plan Pointings window.



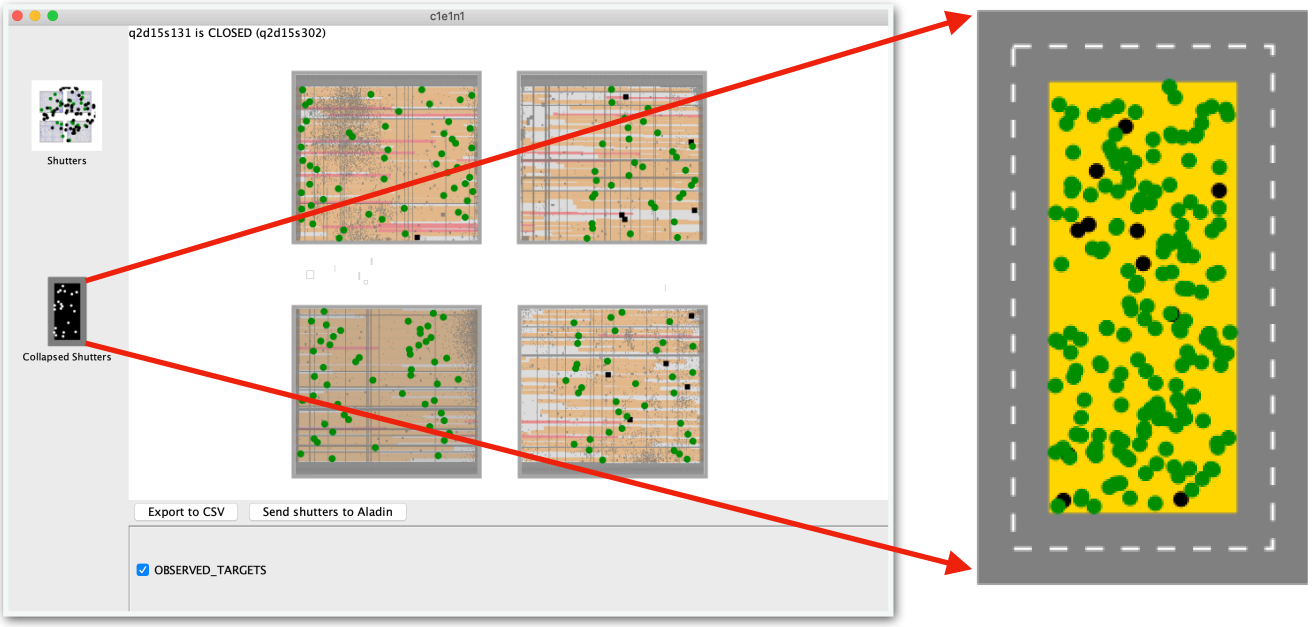
The MSA shutter view of the current configuration highlights finally observed targets in green, and the input targets that were left out of the final configuration in black.



Step 3: If the desired final slit configuration of the current MSA Observation should identically match that output by the eMPT — i.e., it has suffered an undesired (but expected) loss and/or positional shift of targets after import into the MPT system — the *shutter_mask.csv* file should be directly imported into the MPT to overwrite the existing MPT version of the imported eMPT MSA configuration. This is done by returning to the Form Editor view of the current Observation, selecting the *Edit Configuration -> Edit* option in each of the three nodded pointing rows, and manually entering the appropriate *shutter_mask.csv* file in the pop-up window via the Import CSV option.



The superceding eMPT slit configuration may then be inspected and compared to the original 'MPT view' of that same imported configuration — where, e.g., any lost targets in the latter will reappear (in black) and be included in the final observing plan.



4.3 Validate an APT/MPT Plan updated with an eMPT MSA configuration

Once the pre-computed, optimal eMPT MSA pointing summary and slit configuration has been used to update an MPT Plan according to the steps in Section 4.2, the APT ingestion of it may be checked for validity against the MSA field and shutter views output by the *m_pick* module for that optimal configuration.

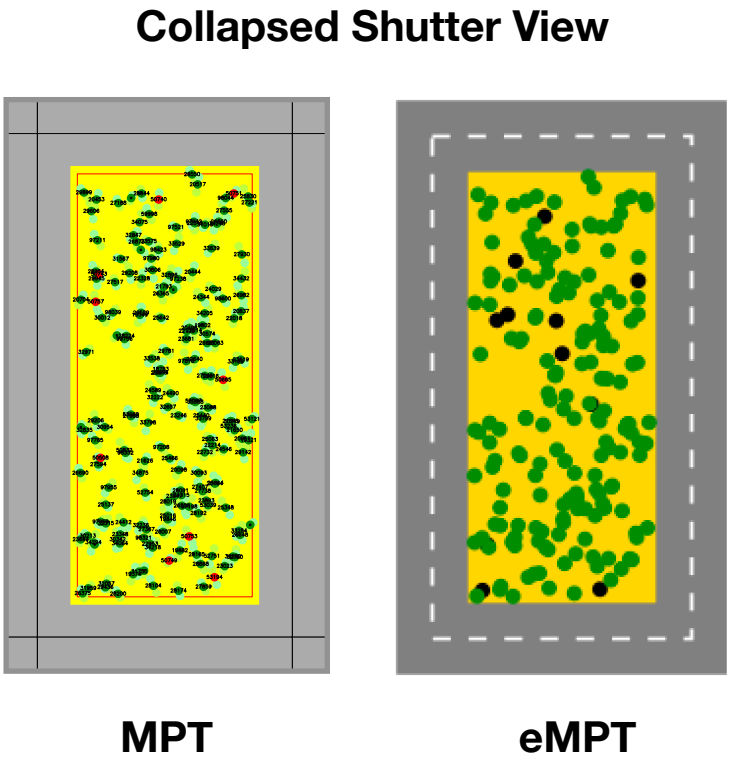
4.3.1. Checking for missing targets from the final MPT MSA configuration

The first verification exercise involves checking that the total number of targets "in at least one exposure" shown in the MSA Planning Tool Plans tab matches the total number expected from the imported eMPT *observed_targets.list*. A small loss of observed targets from the imported eMPT configuration after import into the MPT system —visible in the comparison of Collapsed Shutter views generated by Step 3 — can be due to the different handling of PRISM horizontal spectral overlap between the MPT and eMPT, and/or to the differential velocity aberration assumed.

One possible remedy is to determine the 'angle to target' the MPT assumes and replace the 90 degrees value (i.e. no velocity aberration) assumed in the configuration file of the corresponding eMPT trial with this angle, and re-run the eMPT anew. If this does not resolve the discrepancy, or if there is a significant loss of imported observed targets and/or they have unexpected shutter assignments and intra-shutter locations, proceed to the next verification exercise.

4.3.2. Checking for geometrical inconsistencies between the MPT and eMPT

Both the MPT and eMPT are based on the same NIRSpec instrument model that determines the precise projection of target locations on the MSA plane, therefore the final slit configurations of both systems must be geometrically consistent. An MPT MSA plan curiously missing or showing offset targets from the entered eMPT observed target list may be visualized and further investigated by comparing its MSA Collapsed Shutter view with the corresponding eMPT *collapsed_shutter.ps* plot.



If a comparison of the Collapsed Shutter views are closely similar, this demonstrates that a) a custom eMPT configuration can be imported into the MPT and b) the current eMPT instrument model is geometrically consistent with the independently coded MPT version of the same. In the unlikely event that this test fails, the authors of this User Guide should be contacted for assistance.

APPENDIX

A. *Detailed descriptions of all output summary and diagnostic files of all eMPT modules is provided below for reference.*

1) *ipa* module

a) ./trial_id/ipa_output/map_optimal_pointings.ps - Map showing the relative locations of the optimal pointings projected to the sky.

Pointings achieving the maximum possible Priority Class 1 target coverage are plotted in red, maximum minus one pointings in green, and maximum minus two pointings in yellow. Each pointing is labeled by its unique *ipa*-designated pointing number and can be viewed by zooming in on the plot. The orientation of the plot is in MSA X and Y. The underlying light grey grid showing the MSA shutter pitch is for illustrative purposes only, but serves to aid the user in gauging roughly how many shutters apart two optimal points are in X and Y when evaluating whether a given pair or triple set of pointings constitutes a valid set of suitably offset dithered spectra on the NIRSpec detector. All users employing dithered exposures should familiarize themselves with this map for the observation at hand before proceeding to the next step (each case will be different!).

b) ./trial_id/ipa_output/grouped_optimal_pointings.txt - The list of pointings identified by the *ipa* module grouped by unique sets of targets covered.

It is important to appreciate that if two pointings appearing in the above list and map are designated as covering the same optimal *number* of Priority Class 1 targets, this does not guarantee that the *same set* of targets is covered in both cases. To help delineate this when dithering is employed, the pointings identified by the *ipa* module have in this list been grouped according to which set of targets each group covers. Given the rapid drop-off of the MSA autocorrelation function, groups of pointings covering the same target set will of necessity tend to appear clustered closely together in the map above. Conversely, the farther apart two optimal pointings are on the sky, the fewer targets they are likely to cover in common. Again, users aiming to employ dithered exposures will do well to familiarize themselves with this grouped pointing list before proceeding to the next step.

c) ./trial_id/ipa_output/individual_optimal_pointings.txt - The ungrouped listing of all optimal pointings identified by the *ipa* module, including the maximum-minus-one and -two coverage pointings in addition to the pointings that cover the maximum number of Priority Class 1 targets.

This table provides the details of each individual pointing output by the *ipa* module, including the intra-target locations of the Priority Class 1 targets covered, and their mutual spread in units of MSA X and Y shutter facets. Most users will normally not need to consult this file.

d) ./trial_id/ipa_output/digital_peak_map.ps - Visualization of the primary output of the Initial Pointing optimization Algorithm (*ipa*), a 'digital' map in which each 'pixel' represents the 2-D translation (or "shift vector") along the sky-projected MSA X and Y dimensions from the nominal pointing to the corresponding optimal pointing.

The maxima in this map represent the locations in the shift vector space where pointings of maximal Priority Class 1 targets reside, and serve as seeds for the *ipa* module's second fine-tuning stage in which elimination of contaminated and spectrally overlapping Priority Class 1 targets occurs.

e) ./trial_id/ipa_output/on_sky_map_start.ps - Map showing the locations of the Priority Class 1 targets identified in the user-supplied input catalog projected to the sky (plotted in red) together with the projected centers of all Viable Slitlets for the specified NIRSpec observing mode, and the nominal pointing plotted in green.

Each target is identified by its running number in the input catalog. When repointing the JWST telescope, all targets will move in concert within their surrounding blue search area boxes defined by the specified maximum allowable pointing shift. Only targets containing available green slitlets within their blue search boxes are potentially observable, albeit not necessarily simultaneously.

g) ./trial_id/ipa_output/input_catalog_on_sky.ps

Plot showing the locations on the sky of *all* targets contained in the input catalog, overlaid by the MSA field of view in the *nominal* pointing and the 3-arcminute boundary centered on this position. The input catalog targets are color-coded with Priority Class 1 targets drawn in red and the lower priority classes following a rainbow coloring sequence. Priority Class 0 or negative Priority Class targets included in the input catalog as possible contaminants are shown in gray, and are not to be observed. This color scheme is used throughout the eMPT. The primary purpose of this plot is to verify the soundness of the input catalog, and gauge the uniformity of its target density across the NIRSpec field of view.

2) *k_make* module

./trial_id/k_make_output/k_list_raw.txt - This so-called raw k_list file lists the (integer) target ids, Priority Classes, MSA slitlet designations and (real) relative intra-shutter target locations for all targets located within Viable Slitlets (as defined by the observing mode and Acceptance Zone set for each pointing in the list).

3) *k_clean* module

a) ./trial_id/k_clean_output/k_list_mod.txt - The modified k-list file, which is identical to the *k_list_raw.txt* file output by the *k_make* module, except that the Priority Class designations of all targets found to be contaminated at each pointing have had their signs turned negative such that those targets will be excluded from further consideration by the downstream *m_make* module and therefore not be observed.

4) *m_make* module

The so-called m-list file listing the accepted targets and matching slitlets for each processed single/pair/triple pointing, taking on a slightly different format (and size) depending on the choice of n_dither. This is reflected in the naming of the output m-list file.

./trial_id/m_make_output/single_m_list.txt (n_dither=1)

For n_dither=1, the format of single_m_list.txt closely follows that of the k-list, but listing the final selected targets only. In this case, each analyzed pointing retains the identification number originally assigned to it by the *ipa* module.

./trial_id/m_make_output/pair_m_list.txt (n_dither=2)

./trial_id/m_make_output/triple_m_list.txt (n_dither=3)

For n_dither=2 or n_dither=3 each analyzed pointing pair or triple is assigned a unique identification number in sequence by the *m_make* module, and the individual correlated m-lists for the pointings making up each pair or triple are grouped together in sequence in the combined m-list file.

5) ***m_sort*** **module**

a) Figure-of-merit ranked pointings, depending on choice of n_dither:

i) ./trial_id/m_sort_output/single_list_fom2.txt (n_dither=1)

Listing of the pointings specified in the m_list file ranked according to their figure-of-merit FOM2 (weighted total number of targets observed). Each pointing in the list is identified by its *ipa* pointing designation.

ii) ./trial_id/m_sort_output/pair_list_fom1.txt (n_dither=2)
./trial_id/m_sort_output/pair_list_fom2.txt

Listing of the legal pointing pairs contained in the m_list file output by the *m_make* module, ranked according to the figure-of-merit FOM1 (weighted average exposure time per observed target) *or* FOM2 (weighted total number of targets observed), calculated from the *combined* final target list of the pair. Each legal pointing pair in the list is identified by the pair number assigned to it by the *m_make* module.

iii) ./trial_id/m_sort_output/triple_list_fom1.txt (n_dither=3)
./trial_id/m_sort_output/triple_list_fom2.txt

Listing of the legal pointing triples contained in the m_list file ranked according to the value of their figure-of-merit FOM1 (weighted average exposure time per observed target) or FOM2 (weighted total number of targets observed), calculated from the combined observed target list of the three pointings of the triple. Each pointing triple is identified by the triple number assigned to it by the *m_make* module.

6) ***m_pick*** **module**

Depending on the choice of n_dither, the output of the *m_pick* module specific to each pointing of the dither is contained in the relevant sub-directory ./trial_id/pointing_nn/, ./trial_id/pair_mm/pointing_nn/, or ./trial_id/triple_mm/pointing_nn/ as appropriate.

Note that this directory structure is designed to allow the user to run *m_pick* separately on other potentially interesting solutions in the list without overwriting the detailed output of the previous runs for purposes of comparison.

Each triple/pair/pointing subdirectory contains the following files:

6a) ./trial_id/m_pick_output/pointing_nn/pointing_summary.txt (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/pointing_summary.txt (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/pointing_summary.txt (n_dither=3)

Primary summary file providing the NIRSpec pointing coordinates of the central pointing (and its two nodded pointings) for entry into the STScI APT/MPT, the distribution of the observed targets sorted by Priority Class, the detailed shutter assignments and intra-shutter locations for the observed targets in all three nods, optionally followed by the central shutter designations of any sky background slitlets placed by the *m_pick* module.

6b) ./trial_id/m_pick_output/pointing_nn/observed_targets.cat (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/observed_targets.cat (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/observed_targets.cat (n_dither=3)

Catalog of the targets observed at the pointing, listing each target's running number in that catalog, its running number in the eMPT-formatted sub-catalog, its running number in the master catalog that the sub-catalog was originally extracted from, its Priority Class, and its Right Ascension and Declination in decimal degrees

6c) ./trial_id/m_pick_output/pointing_nn/shutter_mask.csv (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/shutter_mask.csv (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/shutter_mask.csv (n_dither=3)

The STScI-formatted encoded configuration file describing the MSA configuration of the pointing, i.e. which shutters are to be commanded open and which are to remain closed. This file is intended for manual entry into the STScI APT/MPT.

6d) ./trial_id/m_pick_output/pointing_nn/collapsed_shutter.ps (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/collapsed_shutter.ps (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/collapsed_shutter.ps (n_dither=3)

"Collapsed Shutter" plot similar to that available in the STScI MPT, showing the superimposed detailed locations of all observed targets within their slitlets. The specified Acceptance Zone is outlined in red. Dark green points (red for the Priority Class 1 targets) show the intra-shutter location of each observed target at the central pointing, the two lighter green points the slightly shifted locations in the two nodded exposures caused by the differential optical distortion. Targets are labeled (zoom in!) by their running number in the input target catalog.

6e) ./trial_id/m_pick_output/pointing_nn/fov_on_sky.ps (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/fov_on_sky.ps (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/fov_on_sky.ps (n_dither=3)

6f) Even though the resulting final MSA mask at each pointing has only been optimized for the specified disperser, the *m_pick* module produces detailed plots of the target spectral traces for all seven NIRSpec dispersers for the mask produced. It is left to the user to decide which traces are relevant to the program at hand:

./trial_id/m_pick_output/pointing_nn/PRISM_spectra.ps (n_dither=1)
./trial_id/m_pick_output/pointing_nn/G140M_spectra.ps

./trial_id/m_pick_output/pointing_nn/G235M_spectra.ps
./trial_id/m_pick_output/pointing_nn/G395M_spectra.ps
./trial_id/m_pick_output/pointing_nn/G140H_spectra.ps
./trial_id/m_pick_output/pointing_nn/G235H_spectra.ps
./trial_id/m_pick_output/pointing_nn/G395H_spectra.ps

./trial_id/m_pick_output/pair_mm/pointing_nn/PRISM_spectra.ps (n_dither=2)
./trial_id/m_pick_output/pair_mm/pointing_nn/G140M_spectra.ps
./trial_id/m_pick_output/pair_mm/pointing_nn/G235M_spectra.ps
./trial_id/m_pick_output/pair_mm/pointing_nn/G395M_spectra.ps
./trial_id/m_pick_output/pair_mm/pointing_nn/G140H_spectra.ps
./trial_id/m_pick_output/pair_mm/pointing_nn/G235H_spectra.ps
./trial_id/m_pick_output/pair_mm/pointing_nn/G395H_spectra.ps

./trial_id/m_pick_output/triple_mm/pointing_nn/PRISM_spectra.ps (n_dither=3)
./trial_id/m_pick_output/triple_mm/pointing_nn/G140M_spectra.ps
./trial_id/m_pick_output/triple_mm/pointing_nn/G235M_spectra.ps
./trial_id/m_pick_output/triple_mm/pointing_nn/G395M_spectra.ps
./trial_id/m_pick_output/triple_mm/pointing_nn/G140H_spectra.ps
./trial_id/m_pick_output/triple_mm/pointing_nn/G235H_spectra.ps
./trial_id/m_pick_output/triple_mm/pointing_nn/G395H_spectra.ps

The spectra in these plots trace the top, bottom and center of the three shutters making up the slitlet of each target. The target spectra are color-coded by Priority Class and labeled by their running number in the input catalog. The single-shutter spectra from the (currently 19) failed open shutters are drawn in bright pink. The unlabeled spectral traces from the (optionally) added sky background slitlets are shown in gray.

6g) ./trial_id/m_pick_output/pointing_nn/slitlet_usage_stats.txt (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/slitlet_usage_stats.txt (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/slitlet_usage_stats.txt (n_dither=3)

Summary of the global MSA shutter usage statistics at different stages of the target placement process. Casual users need normally not concern themselves with this file (except perhaps to verify that there are indeed no spare Viable Slitlets remaining on the MSA in cases where the *m_pick* module has been tasked with filling out the MSA mask with sky background slitlets).

6h) ./trial_id/m_pick_output/pointing_nn/slitmap_start.ps (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/slitmap_start.ps (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/slitmap_start.ps (n_dither=3)

Detailed map of the entire MSA prior to target placement, with each shutter color-coded according to its status (zoom in!). Shutters filled-in with black are either permanently failed closed or vignetted by the NIRSpec Field Stop. Red filled-in shutters are permanently failed open. Shutters

marked with darker green dots denote the central shutters of fully functional three shutter tall Viable Slitlets. Shutters marked in lighter green dots denote the central shutters of three shutter tall slitlets for which, respectively, one or two of its shutters are non-functional (these are not used in the present eMPT). Shutters marked with red dots cannot serve as the central shutters of Viable slitlets due to their spectra colliding with those of a failed open shutter. Slitlets with central shutters marked with gray dots are similarly excluded from use due to their spectra projecting onto the the detector gap (PRISM only).

6i) ./trial_id/m_pick_output/pointing_nn/slitmap_end.ps (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/slitmap_end.ps (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/slitmap_end.ps (n_dither=3)

Same as 6h) above, but after target placement. This plot is similar to one available in the STScI MPT "show" view. Filled-in shutters marked in blue are the commanded open ones. Shutters marked with orange dots denote the slitlets that are eliminated from use by the successfully placed targets present in the commanded open slitlets.

6j) ./trial_id/m_pick_output/pointing_nn/target_list.txt (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/combined_target_list.txt (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/combined_target_list.txt (n_dither=3)

When n_dither=2 or n_dither=3, this file provides a summary of the COMBINED observed target catalog resulting from the multiple dithered exposures. It begins with a table listing for each target observed its running number in the combined target list, its running number in the eMPT-formatted sub-catalog, its running number in the master catalog from which the sub-catalog was extracted, its priority class, followed by a truth table indicating which of the dithered exposures the target is observed in. This listing is then followed by a detailed breakdown of the combined target list by Priority Class and target exposure time, ending with a listing of the average exposure time in units of sub-exposures received by the targets in each Priority Class.

When n_dither=1, this file merely repeats the information on the (single) observed target list given in (5b-i).

6k) ./trial_id/m_pick_output/pointing_nn/sky_shutters.txt (n_dither=1)
./trial_id/m_pick_output/pair_mm/pointing_nn/sky_shutters.txt (n_dither=2)
./trial_id/m_pick_output/triple_mm/pointing_nn/sky_shutters.txt (n_dither=3)

Separate listing of the central shutters of the empty sky slitlets added to the slitmask.

7) ***m_check*** **module**

./trial_id/m_check_output/pointing_nn/slitlet_panel_plot.ps (n_dither=1)
./trial_id/m_check_output/pair_mm/pointing_nn/slitlet_panel_plot.ps (n_dither=2)
./trial_id/m_check_output/triple_mm/pointing_nn/slitlet_panel_plot.ps (n_dither=3)

Plot showing schematic 1.6 arcsec times 3.7 arcsec close-up maps of the three nodded slitlets of all observed targets listed in the output catalog for the pointing(s) selected by the *m_pick* module. Each panel is labeled by target being observed, and the targets plotted are color-coded by Priority Class and labeled by their running number in the input catalog. The purpose of this plot is to assist the user in determining whether any targets are contaminated by other sources at the specified roll angle to a degree that they should not be observed. Such contaminated sources can be excluded from being considered for observation by setting their Priority Classes negative in the input catalog and running the *ipa* module anew. This MSA map has the same orientation as that available in the "Show" option of the STScI MPT.