Описание программного пакета для фотометрического определения красного смещения.

1. Назначение и состав пакета.
Программный пакет SED, реализованный на языке IDL предназначен для фотометрического определения красного смещения для различного типа внегалактических объектов. Пакет тестировался под операционными системами Linux и Windows 95/98/NT/2000 на платформе Intel x86 в среде RSI IDL версий 5.0 и выше (была сделана модификация пакета для работы с версией 4.0 под Windows).
1.1 Основные преимущества данного пакета.
Главным преимуществом является переносимость кода и аппаратная независимость. Поскольку RSI IDL является интерпретируемым языком программирования, в программе активно используются быстрые функции работы с массивами, что сильно увеличивает производительность при увеличении количества входных данных.
1.2 Недостатки пакета.
К недостаткам следует отнести довольно медленную работу тех участков программы, в которых используются циклы. Также, пожалуй, основным недостатком является отсутствие обработки исключительных ситуаций вычислительнвми функциями (будет реализовано в следующей версии). При использовании IDL 4.0 и более ранних версий возможны проблемы, связанные с длинными именами файлов (более 8 символов).
1.3 Состав пакета.
В настоящее время пакет хорошо структурирован и включает следующие каталоги:
SED
 —__1) ./filters/ - наборы кривых пропускания фильтров
 —__2) ./postscript/ (optional) - результаты работы пакета
 —__3) ./sed.lib/ - программная часть для IDL 5.0 и выше
 —__4) ./sed4.lib/ - программная часть для IDL 4.0
 —__5) ./templates/ - библиотека template-спектров
 —__6) ./teststars/ (optional) - тестовая выборка звезд
 —__7) ./tmp_fluxes/ - вычисленные потоки от объектов в различных
 — светофильтрах
 —____ ./startsw.pro - Start-up файл для sed4.lib
 “____ ./startsed.pro - Start-up файл для нормальной библиотеки

Каталоги 1), 2) и 7) имеют подкаталоги, названия которых соответствуют системам фильтров.
Каталоги sed*.lib имеют подкаталоги addons - дополнительные модули, используемые преимущественно для создания тестовых выборок объектов и создания postscript-файлов.
Назначение каждого модуля и список ключевых слов кратко поясняется в начале каждого файла.

2. Start
Перед стартом отредактируйте файл startsed.pro (startsw.pro) прописав необходимые пути.
Для старта пакета необходимо запустить IDL с рабочей директорией SED (не sed.lib) и написать в приглашении @startsed или @startsw при использовании IDL 4.0 for Windows. При этом будут откомпилированы необходимые модули и добавлены необходимые пути.
Дальнейшие действия зависят от того, что необходимо вычислять.

ПРИМЕЧАНИЕ: при смещении спектра по z вводится попрака за Ly-forest согласно Madau, 1995

2.1 Минимизация chi^2 без использования заранее вычисленных потоков.
Если необходимо определить z для объекта к которому не посчитаны потоки от template-спектра для необходимых значений z или набора фильтров, необходимо набрать следующую команду в приглашении IDL:

```
sed,sed_file='«template spectrum file»',z_range=[«z_min»,«z_max»], «z_step», $
   filter_desc='«filter description file»',fluxes_file='«observed fluxes»', $
```

no_save=1

«template spectrum file» - имя файла, содержащего template-спектр в виде таблицы из двух колонок - длина волны и интенсивность.
«z_min», «z_max», «z_step» - 3 числа, означающих соответственно минимальное, максимальное значения красного смещения и шаг.
«filter description file» - файл с описанием фильтров в следующем формате:
первая строка - число фильтров
дальнешие строки - имена файлов, в которых кривые пропускания хранятся в текстовом виде (2 колонки) - длина волны и пропускание
«observed fluxes» - текстовый файл, содержащий наблюдаемые значения потоков и их ошибки в следующем формате:
первая строка - 2 целых числа - общее число объектов и количество фильтров.
дальнейшие строки в следующем формате
ObjID Flux1 Flux1err Flux2 Flux2err ... FluxN FluxNerr
ObjID - целое число - идентификационный номер объекта
FluxN и FluxNerr - число двойной точности - наблюдаемые значения потоков и их ошибки
no_save=1 - ключевое слово указывает на то, что не нужно считать критерий значимости для каждого значения chi^2. Если данное ключевое слово не установлено или установлено не в 1, то должен существовать подкаталог ./outputdir и там будет сформировано количество файлов равное числу объектов в файле «observed fluxes». Каждый файл будет содержать 3 колонки - значение z, chi^2 и критерий значимости (вероятность)

В этом случае потоки будут посчитаны перед произведением chi^2 минимизации, а выходным результатом будет файл .output_data в текущей директории, куда для каждого объекта будет записано наиболее вероятное красное смещение и соответствующая ему вероятность. Если вероятность = 0, будет записано красное смещение -1.0

2.2 Создание template-потоков для ускорения дальнейшей работы

Если постоянно производить расчет для определенного типа объектов в заданном наборе фильтров, то имеет смысл записать в файл значения потоков, которые получатся при использовании template-спектра и смещении его по z, чтобы не считать их каждый раз. При этом формат вызова следующий:

```
sed,sed_file='«template spectrum file»',z_range=[«z_min»,«z_max»], «z_step», $
  filter_desc='«filter description file»',fluxes_file='«observed fluxes»', $
  tmp_flx_file='«template fluxes file»', no_fit=1
```

«template fluxes file» - имя файла, куда будут записаны вычисленные значения потоков
no_fit=1 - ключевое слово, указывающее на то, что chi^2 минимизацию производить не нужно, необходимо только загрузить объекты.
Если в дальшейшем понадобится произвести минимизацию chi^2, необходимо вызвать процедуру

```
find_z_list,no_save=1,output_name=«file instead .output_data»
```

где no_save=1 означает то же, что в пункте 2.1 а необязательный параметр output_name задает имя файла, куда будет записан результат вместо .output_data .

2.3 Минимизация chi^2 с использованием заранее вычисленных потоков.

Если необходимо определить z для объекта к которому уже вычислены потоки от template-спектра для необходимых значений z или набора фильтров, необходимо набрать следующую команду в приглашении IDL:

```
sed,sed_file='«template spectrum file»',z_range=[«z_min»,«z_max»], «z_step», $
  filter_desc='«filter description file»',fluxes_file='«observed fluxes»', $
  tmp_flx_file='«template fluxes file»',no_save=1
```

«template fluxes file» - имя файла, откуда будут загружены заранее вычисленные значения потоков,
остальные параметры имеют то же значение, что в разделе 2.1 .

2.4 Использование звездных спектров в качестве templates.
В пакет включена библиотека звездных спектров Pickles
При этом для формирования файла с template-потоками необходима специальная программа, и она присутствует в каталоге со спектрами.
При использовании звездных спектров каждому спектру присваивается фиктивное значение z и если оно определяется, следовательно объект может быть отождествлен со звездой соответствующего спектрального класса
Файлы соответствия присутствуют в каталоге tmp_fluxes. stars_numbs содержит номера спектров в файлах, а stars_info_z - соответствующие значения z для потоков, присутствующих в библиотеке.

3. Использование стандартных функций
В этом разделе описывается использование стандартных функций библиотеки. Краткое описание всех стандартных функций можно найти в начале каждого модуля, там разъясняется смысл ключевых слов.

3.1 find_z_obj,«object #»
Функция предназначена для определения красного смещения для конкретного объекта из списка.
При ее вызове производится минимизация chi^2 и вычисление функции вероятности. В результате на экран будет выведен спектр, смещенный по z, значения наблюдаемых потоков с боксами ошибок и template-потоки. При этом используется наиболее вероятное значение z.

3.2 Plot-функции
Программный пакет включает в себя большое число функций построения различных зависимостей.
Краткое описание их можно найти в файле sed.lib/plot_all.pro

3.3 read_fluxes,«filename»
Используется для загрузки наблюдаемых потоков от объектов. Рекомендуется использовать, если нужно заменить список объектов, не меняя остальных данных. Если «filename» не указано, появится диалог открытия файла.

3.4 compute_flux,«z»
Используется для вычисления template-потоков для спектра, смещенного соответственно значению z и с учетом Ly-forest поглощения.

3.5 readsed,«filename»
Используется для загрузки template-спектра в common-область rawspec из файла. Если имя не указано, то появится диалог открытия файла.

3.6 zcomp,«z» и lyalpha_forest.
Используются для смещения по z и введения Ly-forest поглощения в template-спектр. Результат хранится в common-области zcompspec

4. Дополнительные функции.
Дополнительные функции включают в себя программы по формированию списков тест-объектов с заданным отношением сигнал-шум (write_quasars), программы по построению зависимостей разнообразных эффектов селекции от красного смещения и программы по созданию postscript-файлов с примерами спектров и SED для различных объектов и красных смещений.
Все программы, используемые для построения зависимостей эффектов селекции от z, за исключением qsostar, используют в качестве входного файла .output_data (формат описан выше), qsostar использует 2 файла - .f1 и .f2, представляющие собой просто списки вероятностей для каждого объекта, причем первый файл - вероятности при минимизации chi^2 по template-потокам от реального типа объектов (квазар и т. д.), а второй - от звезд.
При этом минимальное значение z и количество объектов для каждого z необходимо задавать в тексте программы.
Еще одно неудобство состоит в использовании device 'X', который, очевидно при работе с Windows следует устанавливать в 'WIN' (фукнции set_plot)

5. Common-области, используемые в программе.
Практически весь обмен данными в программе между модулями производится через

common-области, поэтому здесь приводится их полный список и краткое описание.

```
common obs_fluxes,oflux_n        - наблюдаемые потоки для текущего объекта
common obs_fl_err,ofluxe_n       - их ошибки
common obs_flux_arr,oflux_arr    - весь массив наблюдаемых потоков
common obs_fluxe_arr,ofluxe_arr - весь массив их ошибок
common filters,filt_arr,filt_num- массив кривых пропускания и число фильтров
common norm_k,a_koeff            - нормировочный коэффициент для chi^2
common n_flx,n_fluxes           - количество потоков для текущего объекта
common obj_n,n_obj               - число объектов
common obj_IDS,object_IDS        - идентификатор текущего объекта
common z_par,z_rangev,z_stepv    - параметры вычисления chi^2 по z
common P_arr,pchi2_arr           - Probability function array
common flt_pos,filter_pos        - положения центров фильтров
common flt_wdth,filter_width     - их эквивалентные ширины
common hi_arr,chi2_arr           - chi^2 array
common mfl_arr,mflux_arr         - массив template-потоков
common tmpspec,spec_tmp          - template спектр в исходном виде
common rawspec,spec,i            - он же после интерполяции с шагом 1 A
common zcompspec,zspec           - спектр, смещенный по z и с учетом
                                   Ly-forest поглощения.
```

6. Схема работы.
Составить блок-схему работы практичеки невозможно, поскольку большинство модулей работает независимо.
Возможно лишь пояснение процедур создания template-потоков и минимизации chi^2.

6.1 Создание template-потоков.
При создании template-потоков на входе имеется template-спектр, загруженный модулем readsed. Он находится в common блоке rawspec. Кроме того имются кривые пропускания фильтров, загруженные модулем set_filters.
Для каждого z выполняется процедура compute_flux,z, которая сначала выполняет процедуру zcomp,«z», которая производит смещение спектра и учет Ly-forest поглощения путем вызова lyalpha_forest, а затем производдится вычисление потоков в фильтрах от этого спектра.
Полученные потоки записываются в файл процедурой save_model_fluxes.

6.2 Минимизация chi^2
На вход процедуры find_z подаются параметры поиска (диапазон и шаг). Кроме того в common - блоках хранятся наблюдаемые потоки и их ошибки и template-потоки. Вызывается функция chi2_comp_a, которой на вход подаются эти данные. На первом этапе для каждого набора template-потоков по z вычисляется нормировка вызовом функции find_koeff_a. Получается массив значений нормировочных коэффициентов, который используется при вычислении массива chi^2 значений. На выходе функция chi2_comp_a выдает этот массив. Затем, если установлено ключевое слово comp_prob, find_z выполняет процедуру criter, которая выдает массив критериев значимости, соответствующих данным значениям chi^2. После этого функцию вероятности можно построить процедурой plot_pchi2. Далее определяется минимальное значение chi^2 и вычисляется соответствующая ему вероятность и определяется z, которому она соответствует. Эти данные (z и вероятность) выдаются на выход в файл .output_data (или установленный, LUN 13) либо на экран. Это зависит от значения ключевого слова print_output.