

Стандартная система редукции астрономических данных MIDAS

Лекция III. Возможности графики и визуализации

А.Ю.Князев

Специальная астрофизическая обсерватория РАН, п.Нижний Архыз, 357147, Россия

...ничто рядом с третьей премудростью, сияние которой можно сравнить только с ослепительным блеском солнца и глубину которой можно сравнить только с глубиной океана.

Леонид Соловьев. "Повесть о Ходже Насреддине"

Пришло время более детально познакомиться с возможностями MIDAS-графики и image-дисплея — в дальнейшем я буду называть это пакетом визуализации. При интерактивной работе в мониторе необходимо достаточно подробно представлять их свойства и возможности, а при создании рисунков для статей при помощи MIDAS эти знания просто жизненно важны.

1. Что такое image и графические дисплеи. Их возможности.

Для визуализации изображений используется механизм дисплейных окон. MIDAS запускает свой дисплей-сервер (IDI), который взаимодействует с X11 и MIDAS-монитором. Все команды в MIDAS, использующие дисплей, работают через этот сервер.

1.1. Image-дисплей

Image-дисплей используется в MIDAS для визуализации изображений. Самая необходимая команда при работе с image-дисплеем — это `LOAD/IMAGE`.

ВНИМАНИЕ : Для корректной работы image-дисплея необходимо иметь работающий X Window в режиме 256 цветов. Правда, в последних версиях MIDAS существует возможность работы и в том случае, когда сервер X Window работает в режиме 24-bit True-color. Однако хочу заметить, что такая работа еще не является гарантированной. Установка режима работы делается в этом варианте командой `INITIALIZE/DISPLAY`. Для ознакомления с особенностями работы в таком режиме читайте `HELP [News]`.

Существует возможность открытия до 10 image-дисплеев (от 0-го до 9-го) одновременно. Команда создания image-дисплея — `CREATE/DISPLAY`. Каждый дисплей может иметь до 12 каналов — это значит, что Вы можете загрузить 12 изображений одновременно в 12-ти разных каналах и рассматривать или работать с ними, указывая номер канала. Число каналов создаваемого дисплея задается в команде `CREATE/DISPLAY` (вместе с размерами дисплея), а номер текущего канала устанавливается командой `DISPLAY/CHANNEL`. Эти каналы могут быть такого же самого размера, как и размер дисплейного окна, но могут быть и больше, и тогда весьма полезна команда `SCROLL/CHANNEL`. Все команды работы с каналами имеют квалификатор `CHANNEL`: `BLINK/CHANNEL`, `CLEAR/CHANNEL`, `DISPLA/CHANNEL`, `SCROLL/CHANNEL`, `SHOW/CHANNEL`, `ZOOM/CHANNEL`.

Дополнительный канал созданного image-дисплея всегда используется как оверлейный (графический) канал. Таким образом, поверх визуализированных изображений можно также рисовать графические картинки, перенаправив вывод с помощью команды `ASSIGN/GRAPHICS`). Для работы с каналом в оверлейном режиме существует многочисленный набор команд, которые я охарактеризую ниже.

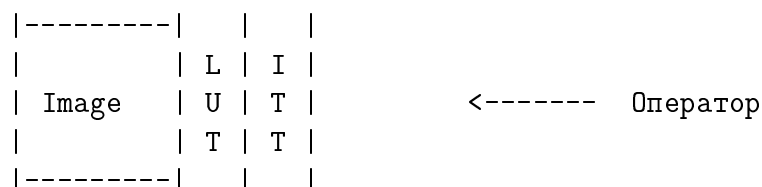
ВНИМАНИЕ : Помните, что все команды переназначения начинают работать *ТОЛЬКО* после создания соответствующих дисплеев и окон.

ВНИМАНИЕ : Хотя разработчики *MIDAS* утверждают, что оверлейный канал создается всегда, я обращаю Ваше внимание на то, что число каналов в созданном image-дисплее должно быть больше 1 для гарантийного использования возможности оверлея. В противном случае *MIDAS* может обрабатывать задаваемые Вами команды, не выводя никаких результатов и ничего не сообщая.

Также для каждого image-дисплея можно создать:

- zoom-окно (команда создания — `CREATE/ZOOM`) — в него выводится увеличенный в несколько раз участок визуализированного изображения. Существуют команды, которые автоматически создают это окно (например, `VIEW/IMAGE`);
- shadow-дисплей (параметр команды `CREATE/DISPLAY`) — параллельный вывод результатов визуализации на любой компьютер в сети, имеющий X Window.

Рассмотрим общую схему image-дисплея, которую я показал на приведенном ниже рисунке. Оператор-астроном смотрит на изображение справа. Image-дисплей позволяет раскрашивать визуализированные изображения в псевдо-цвета (256 цветов максимум), менять их интенсивность. Но вариантов цветовых гамм очень много. За конкретный вариант цветовой гаммы отвечает LUT-таблица (Look-Up Table), которая адаптирует человеческое восприятие цветов к изображенным на экране. Однако есть еще один слой, накладывающий дополнительный закон на LUT-таблицу. Этот слой называется ITT-таблица (Intensity Transfer Table), и его также можно менять.



Рассмотрим теперь подробнее различные наборы команд для работ с image-дисплеем.

ВНИМАНИЕ : Иногда (а для некоторых пользователей, у которых пальцы двигаются по клавиатуре быстрее, чем думает голова — очень часто) пользователь совершает набор движений, после которых связь между MIDAS-монитором и IDI-сервером разрушается. Это выражается в том, что графика перестает работать вообще либо на экране возникает нечто неожиданное, изображения не визуализируются и так далее. В этом случае необходимо помнить о могучей команде **RESET/DISPLAY**, которая восстанавливает эту связь. Только не удивляйтесь, что после выполнения данной команды все графические и image-дисплеи исчезнут — их необходимо создать заново.

1.1.1. LUT — Look-Up Tables

Принцип показа визуализированных интенсивностей различными цветами очень прост: весь их перепад (а его границы определяются параметром **CUTS** команды **LOAD/IMAGE**) разбивается на 256 интервалов, каждый из которых обозначается своим цветом. Поскольку любой цвет определяется сочетанием трех основных: **R** — красного, **G** — зеленого и **B** — голубого, — то LUT-таблица просто определяет для каждого из 265 интервалов интенсивность этих трех. Она изменяется от 0 до 1. Например, если ее изменение одинаково линейно от 0 до 1 для RGB, то цвет будет изменяться от белого к черному, а если наоборот — от 1 до 0, то от черного к белому.

В MIDAS существует большой набор стандартных LUT-таблиц. Их можно модифицировать в интерактивном режиме. LUT-таблицы помогают выделить на визуализированном изображении его особенности различным распределением цвета. Чтобы посмотреть, как выглядит одно и то же изображение при использовании различных LUT-таблиц, выполните команду **TUTORIAL/LUT**. Она же покажет Вам и основные принципы работы с этими таблицами. Все команды работы с ними имеют квалификатор **LUT**. Это: **CLEAR/LUT**, **CREATE/LUT**, **DISPLA/LUT**, **GET/LUT**, **LOAD/LUT**, **MODIFY/LUT**, **SET/LUT**, **TUTORIAL/LUT**.

Замечание : По умолчанию всегда загружается LUT-таблица, соответствующая равномерному распределению цвета от черного до белого. Ее имя — "ramp".

Некоторые из существующих стандартных LUT-таблиц имеют следующие имена: **backgr**, **color**, **heat**, **light**, **pastel**, **pseudo1**, **pseudo2**, **rainbow**, **rainbow1** ... **rainbow4**, **random**, **random1** ... **random4**, **smooth**, **staircase**, **stairs8**.

1.1.2. ITTs — Intensity Transfer Tables

Как я уже говорил выше, при помощи ITT-таблиц делается преобразование текущих значений загруженной LUT-таблицы в соответствии со значениями ITT (накладывается некоторая функция преобразования). Например, можно наложить на черно-белое распределение "ramp" дополнительно логарифмический закон (команда **LOAD/ITT log**), и тогда Вы получите такое же распределение интенсивностей, как на фотопластинке.

В MIDAS существует большой набор стандартных ИТТ-таблиц. Их можно модифицировать в интерактивном режиме. Чтобы посмотреть, как выглядит одно и то же изображение при использовании различных ИТТ-таблиц, выполните команду TUTORIAL/ИТТ. Она же покажет Вам и основные принципы работы с этими таблицами. Все команды работы с ними имеют квалификатор ИТТ. Это: CLEAR/ИТТ, GET/ИТТ, LOAD/ИТТ, MODIFY/ИТТ, SET/ИТТ, TUTORIAL/ИТТ.

Замечание : По умолчанию всегда загружается ИТТ-таблица, соответствующая нулевому дополнительному закону. Ее имя — "gap".

Некоторые из существующих стандартных ИТТ-таблиц имеют следующие имена: **neg, expo, log, neglog, jigsaw, staircase.**

1.1.3. Работа с курсором

Очень часто для работы с визуализированными изображениями необходимо использовать курсор. Каждый image-дисплей может работать с двумя курсорами одновременно (например, EXTRACT/TRACE): один будет управляться "мышью", а второй — клавиатурой. Если курсор управляется "мышью", то левая кнопка ВСЕГДА служит для чтения текущего положения курсора, а правая и средняя — для выхода из программы.

ВНИМАНИЕ : Для чтения текущего положения курсор должен быть неподвижен, что иногда очень трудно сделать при наличии "мыши" плохого качества.

Если курсор управляется с клавиатуры, то направление движения определяется стрелками на клавиатуре, размер шага — клавишами от 1 до 9, а для чтения текущего положения служит клавиша < Enter >.

Очень многие команды используют курсор прямоугольной или круглой формы (например, VIEW/IMAGE). В этом случае каждому курсору соответствует так называемая "область интереса" (ROI — a region of interest). Размер этой области может меняться либо при помощи клавиш (от 0 до 9 на клавиатуре), либо при помощи других, специально оговоренных, клавиш.

Все команды работы с курсором имеют квалификатор CURSOR: GET/CURSOR, CLEAR/CURSOR, SET/CURSOR.

1.1.4. Графика

Как я уже говорил, у image-дисплея существует оверлейный канал, в котором пользователь также может рисовать все то, что он рисует на графическом дисплее (мы будем рассматривать все эти команды далее). Кроме того, существует набор дополнительных команд, позволяющих управлять рисованием в оверлейном канале (разрешать, запрещать, очищать): SET/OVERLAY, CLEAR/OVER, CLEAR/CHAN OVER — и набор команд, которые могут рисовать ТОЛЬКО в оверлейном канале: DRAW/... (CIRCLE, RECTANGLE ...), LABEL/DISPLAY.

ВНИМАНИЕ : Очень часто пользователь путает команду CLEAR/OVERLAY — запрещающую вообще вывод в оверлейный канал, с командой CLEAR/CHANNEL OVERLAY, очищающей оверлейную память.

1.1.5. Алфавитно-цифровая память

Каждое дисплейное окно может иметь (а может и не иметь) так называемую алфавитно-цифровую память, которая создается командой `CREATE/DISPLAY`. Эта память содержит 3 строки и позволяет выводить из нее различные характеристики визуализованного изображения. Команды работы с памятью: `LABEL/DISPLAY`, `CLEAR/ALPHA`.

1.2. Графический дисплей и графический пакет

Существует возможность открыть до 10 графических дисплеев (от 0-го до 9-го) одновременно. Команда создания графического дисплея — `CREATE/GRAPH`. Работа с графическим дисплеем является частью работы с графическим пакетом MIDAS в целом.

Все команды для работы с графическим пакетом MIDAS можно разбить на три основные группы:

Общие команды — это команды создания/уничтожения дисплейных графических окон, а также команды настройки, просмотра состояния и команды ввода-вывода на различные внешние устройства.

- `CREATE/GRAPH`, `DELETE/GRAPH`, `CLEAR/GRAPH` — команды создания, уничтожения и очистки;
- `SET/GRAPH`, `SHOW/GRAPH` — команды настройки и просмотра текущего состояния параметров графического пакета. Команда `SET/GRAPH` является, с моей точки зрения, самой важной для графического пакета, так как позволяет производить все настройки. В Таблице 1 я кратко перечислил все возможные параметры этой команды и их назначение;
- `ASSIGN/GRAPH`, `COPY/GRAPH` — команды перенаправления вывода. Обычно используются для получения копии созданного рисунка;
- `PLOT/AXES`, `OVERPLOT/AXES`, `LABEL/GRAPH`, `OVERPLOT/LINE`, `OVERPLOT/SYMBOL`, `OVERPLOT/GRID` — рисование осей, символов, линий, формул и т.д. Необходимо знать, что, как и в \LaTeX , в графическом пакете MIDAS можно использовать специальные команды для рисования специальных символов, подъема и опускания текста. Я суммировал это в приведенных Таблице 2 и Таблице 3.

В качестве примера использования этих знаний, хочу привести следующий:

Пример: Если написать в MIDAS-мониторе следующую строку

```
LABEL/GRA "e\{\!u(x\{\!u2\}+y\{\!u2\})\}= -
(\alpha+\beta ) sin\{\!u2\}\theta
```

то результат на графическом дисплее будет выглядеть так:

$$e^{(x^2+y^2)} = (\alpha + \beta) \sin^2 \theta,$$

Команды рисования — это команды для создания новых рисунков (`.../PLOT`) на основе различных структур данных (спектров, таблиц ...) или добавления изображения к уже существующему (`.../OVERPLOT`).

Таблица 1. Параметры команды SET/GRAPH

Параметр	Возможные значения и значение "по умолчанию"
DEFAULT	Устанавливает значения всех параметров "по умолчанию"
XAXIS=	пределы и разметка по X-оси. AUTO или xstart, xend, xbig_tick, xsmall_tick в мировых координатах; когда xsmall_tick < 0 рисуется в логарифмической шкале; значение "по умолчанию" — AUTO
YAXIS=	пределы и разметка по Y-оси. AUTO или ystart, yend, ybig_tick, ysmall_tick в мировых координатах; когда ysmall_tick < 0 рисуется в логарифмической шкале; значение "по умолчанию" — AUTO
ZAXIS=	тоже, что для YAXIS
FRAME=	RECT или SQUA; значение "по умолчанию" — RECT
XSCALE=	масштабирование. AUTO, масштаб в мировых units/per mm или размер рисунка
YSCALE=	масштабирование. AUTO, масштаб в мировых units/per mm или размер рисунка
XOFFSET=	NONE или сдвиг от левой границы окна
YOFFSET=	NONE или сдвиг от нижней границы окна
XFORMAT=	Формат подписи по X-оси. NONE, AUTO или описание формата (смотри HELP)
YFORMAT=	Формат подписи по Y-оси.
ZFORMAT=	тоже, что для YFORMAT
Pmode=	-1 — рисование без осей и подписей, 0 — рисование с подписанными осями, 1 — рисование с подписанными осями и некоторой дополнительной информацией, 2 — рисование с подписанными осями и полной информацией ("по умолчанию")
FONT=	назначение фонта, который будет использован при написании текста на рисунке; значение "по умолчанию" — 1; (see below)
LTYPE=	тип линии. От 1 (непрерывная) до 6 (прерывная длинная-короткая); значение "по умолчанию" — 1; 0 соответствует отсутствию соединения точек линиями (для таблиц)
SType=	тип символа: от 1 (точка) до 21 (заполненный ромб); значение "по умолчанию" — 4 (крестик); 0 соответствует отсутствию символа
LWIDTH=	установка ширины рисуемой линии; 0 или 1 это нормальная ширина; 2, 3 и 4 для увеличения толщины (ширины)
SSIZE=	число, устанавливающее масштабный фактор для рисуемого символа; значение "по умолчанию" — 1
TSize=	число, устанавливающее масштабный фактор для текста; значение "по умолчанию" — 1
TWIDTH=	установка толщины текста; 0 или 1 это нормальная ширина; 2, 3 и 4 для увеличения толщины; значение "по умолчанию" — 1; Результат будет виден ТОЛЬКО на PostScript-картинке!
BINMODE=	OFF или ON; "по умолчанию" — OFF
COLOUR=	цвет. От 0 до 7; "по умолчанию" — черный (1);
BCOLOUR=	цвет фона. От 0 до 7; "по умолчанию" — белый (0);
CLEARGRA=	ON или OFF. при OFF графический дисплей не очищается даже при использовании следующей команды PLOT; значение "по умолчанию" — ON

Таблица 2. Использование специальных символов в графическом пакете MIDAS

Metacharacter	Meaning
\{	начало группы
\}	конец группы
\^	поднять на половину высоты символа
\!u	поднять на половину высоты символа
_	опустить на половину высоты символа
\!d	опустить на половину высоты символа
\<	сдвинуть влево на ширину символа
\+	увеличить размер текста на 20%
\-	уменьшить размер текста на 20%
\!	рассматривать следующую за этим последовательность как последовательность метасимволов (это разрешает использовать метапоследовательности, начинающиеся с 'n', не рассматривая их как переход на новую строку)
\0	установить шрифт 0 ("по умолчанию")
\1	установить шрифт 1 (Quality roman font)
\2	установить шрифт 2 (Greek font)
\3	установить шрифт 3 (Script font)
\4	установить шрифт 4 (Old English)
\5	установить шрифт 5 (Tiny roman font)
\[увеличить толщину линий (bolding; optional)
\]	уменьшить толщину (bolding; optional)
\n	переход на новую строку
~~	Нарисовать символ '~'
\~	Нарисовать символ '~'
~\	Нарисовать символ '\'
\\	Нарисовать символ '\'

Таблица 3. Использование \TeX -подобных символов в графическом пакете MIDAS

<code>\AA</code>	<code>\Alpha</code>	<code>\Aquarius</code>	<code>\Aries</code>
<code>\Beta</code>	<code>\Cancer</code>	<code>\Capricorn</code>	<code>\Chi</code>
<code>\Delta</code>	<code>\Earth</code>	<code>\Epsilon</code>	<code>\Eta</code>
<code>\Gamma</code>	<code>\Gemini</code>	<code>\Iota</code>	<code>\Jupiter</code>
<code>\Kappa</code>	<code>\Lambda</code>	<code>\Leo</code>	<code>\Libra</code>
<code>\Mars</code>	<code>\Mercury</code>	<code>\Moon</code>	<code>\Mu</code>
<code>\Neptune</code>	<code>\Nu</code>	<code>\Omega</code>	<code>\Omicron</code>
<code>\PI</code>	<code>\Phi</code>	<code>\Pisces</code>	<code>\Pluto</code>
<code>\Psi</code>	<code>\Rho</code>	<code>\Sagittarius</code>	<code>\Saturn</code>
<code>\Scorpio</code>	<code>\Sigma</code>	<code>\Sqrt</code>	<code>\Tau</code>
<code>\Taurus</code>	<code>\Theta</code>	<code>\Upsilon</code>	<code>\Uranus</code>
<code>\Venus</code>	<code>\Virgo</code>	<code>\Xi</code>	<code>\Zeta</code>
<code>\aleph</code>	<code>\alpha</code>	<code>\asteroid</code>	<code>\beta</code>
<code>\bigcirc</code>	<code>\black</code>	<code>\blue</code>	<code>\cent</code>
<code>\chi</code>	<code>\circ</code>	<code>\cyan</code>	<code>\clover</code>
<code>\clubsuit</code>	<code>\comet</code>	<code>\dag</code>	<code>\ddag</code>
<code>\default</code>	<code>\delta</code>	<code>\diamond</code>	<code>\div</code>
<code>\downarrow</code>	<code>\epsilon</code>	<code>\equinox</code>	<code>\equiv</code>
<code>\eta</code>	<code>\firtree</code>	<code>\gamma</code>	<code>\ge</code>
<code>\greek</code>	<code>\green</code>	<code>\hbar</code>	<code>\heart</code>
<code>\infty</code>	<code>\int</code>	<code>\iota</code>	<code>\italic</code>
<code>\kappa</code>	<code>\lambda</code>	<code>\larrow</code>	<code>\le</code>
<code>\magenta</code>	<code>\mp</code>	<code>\mu</code>	<code>\!nabla</code>
<code>\!ne</code>	<code>\!nu</code>	<code>\odot</code>	<code>\oint</code>
<code>\old</code>	<code>\omega</code>	<code>\omicron</code>	<code>\oplus</code>
<code>\otimes</code>	<code>\palmtree</code>	<code>\paragraph</code>	<code>\parallel</code>
<code>\partial</code>	<code>\perp</code>	<code>\phi</code>	<code>\pi</code>
<code>\pm</code>	<code>\propto</code>	<code>\psi</code>	<code>\red</code>
<code>\rho</code>	<code>\rightarrow</code>	<code>\roman</code>	<code>\script</code>
<code>\shield</code>	<code>\sigma</code>	<code>\snow</code>	<code>\spade</code>
<code>\sqrt</code>	<code>\sum</code>	<code>\tau</code>	<code>\theta</code>
<code>\times</code>	<code>\tiny</code>	<code>\uparrow</code>	<code>\upsilon</code>
<code>\varepsilon</code>	<code>\varphi</code>	<code>\vartheta</code>	<code>\white</code>
<code>\xi</code>	<code>\yellow</code>	<code>\zeta</code>	

PLOT/CONTOUR, PLOT/COLUMN, PLOT/ROW, PLOT/DESCRIPTOR, PLOT/GRAY, PLOT/HISTOGRAM,
PLOT/KEYWORD, PLOT/PERSPECTIVE, PLOT/TABLE OVERPLOT/ERROR, PLOT/VECTOR

Команды для работы с графическим курсором — команды для снятия текущих координат, установления типа курсора и так далее.

GET/GCURSOR, CENTER/GAUSS, MODIFY/GCURSOR, INTEGRATE/LINE, INTEGRATE/STAR,
INTEGRATE/STAR

1.3. Пакет визуализации и создание рисунков

Хорошее знание пакета визуализации в MIDAS (работа с image и графическим дисплеями) избавляет пользователя-астронома практически полностью от необходимости работать с другими внешними пакетами при подготовке рисунков для статей. Вы как бы погружены в единую среду, в который производится как обработка данных (или расчет моделей), так и подготовка к выдаче Ваших результатов "наружу". Конечно, существует ряд рисунков, которые очень трудно создать в MIDAS, и поэтому я не берусь утверждать, что этот пакет есть панацея от всех бед. Но грамотный пользователь очень легко расширяет его возможности путем написания командных файлов.

ВНИМАНИЕ : При создании рисунков для статей **ОБЯЗАТЕЛЬНО** оформляйте их в виде командных файлов: один рисунок — один командный файл. Храните этот командный файл и данные для него в одном месте. При необходимости переделать/доделать рисунок Вы просто слегка модифицируете имеющийся командный файл.

Я хотел бы привести в пример два командных файла с результатами их работы, конечно. Как будет говориться в дальнейшем, командный файл в MIDAS обычно имеет расширение ".prg" и, если он находится в текущей директории, вызывается на исполнение MIDAS-монитором командой @@ ИМЯ_ФАЙЛА. При разборе приведенных примеров учтите, что графический и image-дисплей я уже создал до исполнения командных файлов.

Пример :

Есть изображение **h038Rbffm.bdf** с прямым снимком галактики. Необходимо его нарисовать для статьи, наложив контуры на центральную часть галактики. На рисунке надо показать масштаб. Я написал командный файл, который назвал **2D.prg** и вызвал его командой @@ 2D. Ниже даны краткие пояснения некоторым командам, а полный разбор строки со всеми параметрами оставлен интересующимся. Результат работы данной программы приведен на Рисунке 1.

LOAD/IMA h038Rbffm 0 4 1002,1060 cuts=11300,11650 — визуализуем изображение в черно-белом цвете;

LOAD/ITT neg — устанавливаем ИТТ-таблицу, которая выбирает светлый цвет (белый) для фона и темный (черный) для галактики;

@a vertlut ? black 10 f5.0 — рисуем вертикальный бар с распределением цвета в зависимости от интенсивности (при этом стандартная команда выводит на дисплей (в оверлейный канал) значения уровней);

CLEAR/CHANEL OVER — очищаем оверлейный канал, чтобы этих надписей не было;

ASS/GRA d,0 — переназначаем вывод графики на image-дисплей;

SET/GRAPH tsize=0 ssize=0 — чтобы следующая команда не рисовала ничего лишнего, устанавливаем нулевые размеры для линий и букв;

PLOT/AXES [<,<:>,>] — совмещаем визуализованное изображение с оверлейной памятью;

ВНИМАНИЕ : *Эта команда есть тайное знание, которым я делюсь со всеми! Помните о ней! И всегда используйте, если Вы хотите, чтобы нарисованная контурная карта лежала поверх визуализованного изображения, а не стояла задумчиво в стороне.*

SET/GRA tsize=1 ssize=1 — возвращаем исходные числа;

DRAW/LINE 100,50,200,50 S ? black — рисуем горизонтальный бар. Чтобы он выглядел достаточно "жирным" (мы ведь потом будем рисунок сжимать), проводим три линии с легким сдвигом одна относительно другой. Рисуем командами для работы с оверлеем (а не с графикой), поскольку эти команды работают в единицах экрана, а не изображения (которое я загрузил с увеличением 4), значит сдвиг на один в координатах изображения — это сдвиг на 4 в координатах экрана, и мой трюк с тремя линиями не сработает (можете попробовать);

DRAW/LINE 100,51,200,51 S ? black

DRAW/LINE 100,52,200,52 S ? black

DRAW/LINE 99,46,99,56 S ? black — рисуем вертикальную черту на левом конце бара. Используем тот же самый прием;

DRAW/LINE 100,46,100,56 S ? black

DRAW/LINE 101,46,101,56 S ? black

DRAW/LINE 199,46,199,56 S ? black — рисуем вертикальную черту на правом конце бара;

DRAW/LINE 200,46,200,56 S ? black

DRAW/LINE 201,46,201,56 S ? black

LABEL/DIS "5" 55,145 ? black 2 — рисуем надпись рядом с баром, показывающим масштаб. Работаем в координатах экрана (для однообразия);

LABEL/DIS "" 57,157 ? black 2

LABEL/DIS "" 57,162 ? black 2

LABEL/DIS "303 pc" 25,112 ? black 2

SET/GRA lwidth=3 color=8 — устанавливаем цвет и ширину линий, которыми будем рисовать контуры;

OVERPLOT/CONT h038Rbfff ? 11700,11800:15000:600 LTYPE — рисуем контуры только некоторой центральной (самой яркой) части;

SET/GRA lwidth=1 color=1 — возвращаем исходные числа;

`COPY/DIS p3=P p5=noprint p6=PC8NZ` — копируем получившийся результат в PostScript-файл;

`$mv screen10.ps HS0822+3542R_direct.ps` — переименовываем стандартный файл в нужное имя.

Пример :

Есть таблица **qq.tbl** со значениями измеренных скоростей для различных областей одной галактики и расстояниями для этих областей от центра галактики. Необходимо нарисовать для статьи эти точки, посчитать и наложить на рисунок получающуюся кривую скоростей (линейное приближение). Показать точки, не совпадающие с этой кривой скоростей. Я написал командный файл, который назвал `Rot.prg`, и вызываю его командой `@@ Rot`. Результат работы данной программы приведен на Рисунке 2.

`SET/GRA ssize=1.5 tsize=1.3 font=1 twidth=2 stype=3` — производим настройки графического пакета для рисования линий и символов;

`PLOT/AXES 0,50 950,1250 ? "\+\1 Distance (arcsec)" "\+\1 Velocity (km/s)"` — рисуем оси вместе с подписями;

`SELECT/TAB qq :POS_C.lt.0` — выделяем в таблице все точки по одну сторону от центра галактики;

`OVERPLOT/TAB qq :POS_CRA :VEL_C` — рисуем их пустыми квадратиками;

`SET/GRA stype=8` — меняем тип рисуемых символов на звездочки;

`SELECT/TAB qq :PO_C.gt.0` — выделяем в таблице все точки по другую сторону от центра галактики;

`OVERPLOT/TAB qq :POS_CRA :VEL_C` — рисуем их звездочками;

`SELECT/TAB qq all` — выделяем в таблице все точки;

`OVERPLOT/ERROR qq :POS_CRA :VEL_C :Err` — рисуем бары ошибок для всех точек;

`REGRESSION/POLYNOM qq1 :VEL_C :POS_CRA 1` — я заранее скопировал во вторую таблицу **qq1.tbl** все точки, за исключением двух, которые сильно отклоняются. Используя точки из этой второй таблицы, я приближаю их полиномом первой степени;

`SAVE/REGRES qq1 coef` — запоминаю полученные коэффициенты;

`COMP/REGRES qq1 :fit = coef` — считаю положения точек в соответствии с полученным приближением;

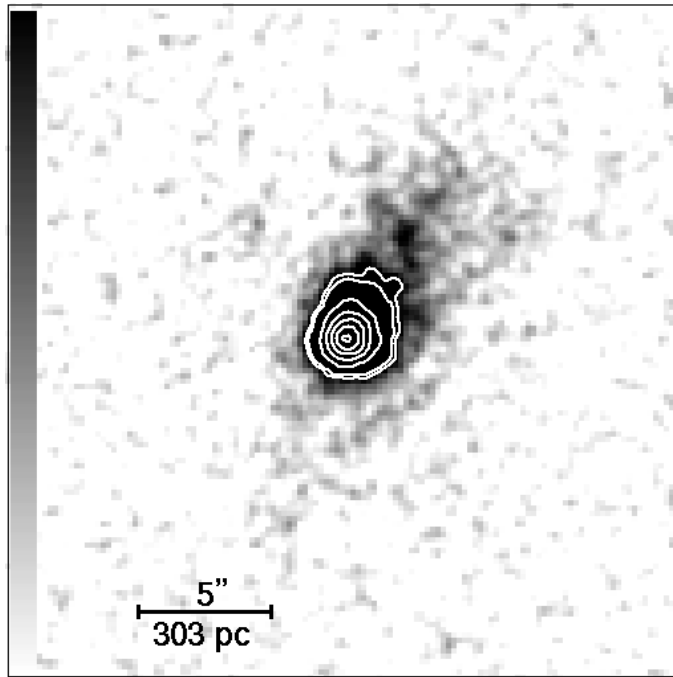


Рис. 1. Изображение галактики HS 0822+3542 с наложенными на его центральную часть контурами. Показан масштаб.

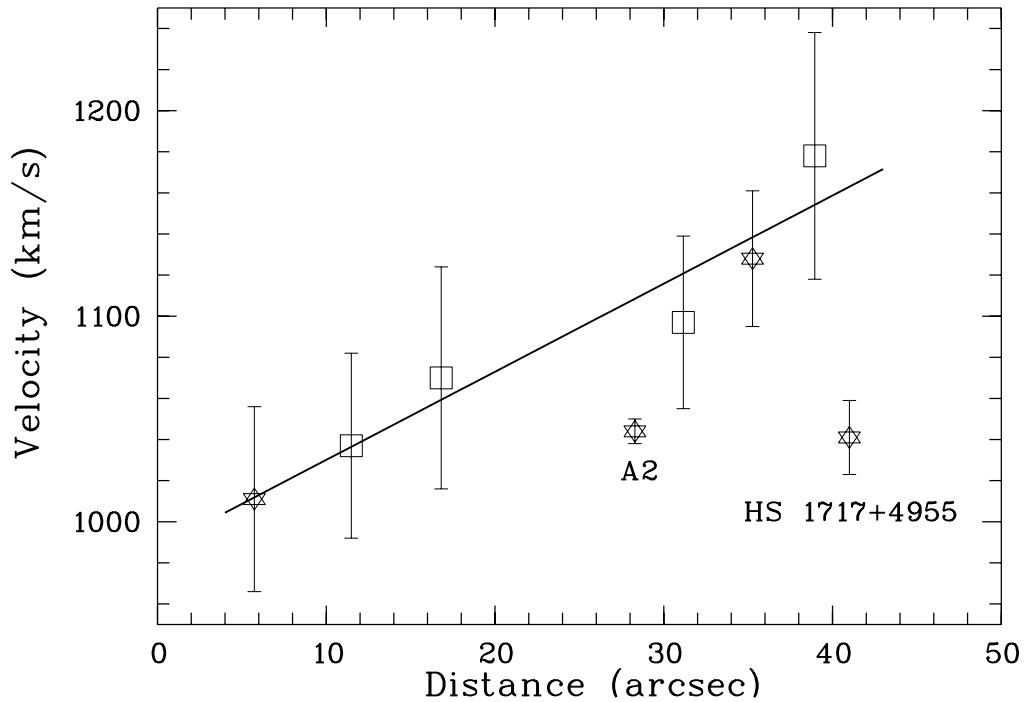


Рис. 2. Кривая вращения для галактики UGC 10806. Звездочками отмечены скорости для областей из удаляющегося, а квадратиками — из приближающегося рукава этой галактики. Вертикальными барами выделены ошибки для каждой точки. Сплошная линия показывает приближение наблюдаемых точек полиномом первой степени. Отмечены положения HS 1717+4955 и области A2, не использованные при построении кривой вращения.

CREATE/IMA ref 1,40 4,1 nodata — это мой "ход конем". Мне не нравится вид прямой, проведенной по шести точкам, когда я просто пытаюсь использовать PLOT/TABLE, установив перед этим SET/GRAPH STYPE=0 LTYPE=1. К тому же MIDAS рисует в этом случае только до положений крайних точек. Поэтому я решил построить модель линейного приближения в виде одномерного изображения. Как первый шаг я строю так называемое reference-изображение (в нем нет данных, но есть размерность, шаг и т.д.);

CONVERT/TABLE vel = qq1 :POS_CRA :fit ref POLY 1 — преобразую колонку таблицы в изображение **vel.bdf**, используя число точек, размерность и шаг;

SET/GRA lwidth=3 — устанавливаю тип линии для рисования;

OVER/ROW vel — разрисовываю построенную модель;

SET/GRA ssize=1 tsize=1 — возвращаю исходные установки для части параметров;

LABEL/GRA "A2" 28.5,1025 ? 1 — делаю надписи, указав место на рисунке;
 LABEL/GRA "HS 1717+4955" 41,1005 ? 1

SET/GRA ssize=1 tsize=1 twidth=1 stype=5 lwidth=1 — возвращаю исходные установки для всех параметров;

COPY/GRA postscript — копирую полученный рисунок в PostScript-файл;

\$mv postscript.ps HS1717_rot.ps — переименовываю стандартный файл.

И это конец третьей сказки о MIDAS.

Список литературы

MIDAS Users Guide, 1995a, **Volume A**

MIDAS Users Guide, 1995b, **Volume B**

Но послушай теперь ... мою ... премудрость, которая стóит всех твоих.

Леонид Соловьев. "Повесть о Ходже Насреддине"