Знакомство с ГИС GRASS

Колесов Д. А., Мурый А. А.

КНИТУ-КАИ, ИГЭ РАН

2012

Содержание

GRASS: Geographic Resources Analysis Support System

- Разрабатывается с 1984 года (USA-CERL). Все это время была открытой ГИС.
- Кроссплатформенная:
  - доступны версии для GNU/Linux, MS-Windows, Mac OSX, SUN, . . . ;
  - 32/64 битные системы.
- Хорошо документирована, большие коллекции данных. Коммерческая поддержка.
- Русское зеркало: http://grass.gis-lab.info/index.php

ГИС GRASS это:

- Растровая 2.5D/3D ГИС
- Векторная 2D/3D топологическая ГИС
- Анализ и обработка графов
- Система обработки изображений
- Система 2D и 3D визуализации
- Поддержка баз данных:
  - DBF, sqlite;
  - PostgreSQL, MySQL;
  - ODBC: MS SQL, Oracle, ...
- Поддерживает все распространенные растровые и векторные форматы.
- Взаимодействие с другими программами:
  - Мат.пакеты, статистика (R, gstat, MatLab/Scilab, ...);
  - Internet-протоколы OGC (WMS, WFS, WCS, WPS);
  - ГИС (QGIS);

Формулировка задачи построения маршрута

Для демонстрации приемов работы в GRASS выбрана следующая задача:

Задача коммивояжера

Есть определенное число точек на местности, которые необходимо посетить. Требуется спланировать маршрут между ними, так, чтобы посетить каждую точку, при этом маршрут должен быть оптимальным по определенному критерию.

Уточним условия:

- Пусть планируется пеший маршрут. Будем искать такой путь, который будет наименьший по времени.
- На скорость движения влияет тип местности (поле/лес, наличие дорог, необходимость перехода рек вброд и т.п.). Будем учитывать рельеф: на крутых склонах скорость движения замедляется.

Построение маршрута: план работы

Для построения оптимального маршрута нам необходимо проанализировать местность. Для этого:

1. Создадим проект и импортируем в него исходные данные:
   - Данные космической съемки (Landsat).
   - Данные о рельефе (Aster GDEM).
   - Векторные данные о местности (OSM).
2. Произведем дешифрирование снимков Landsat и выделим на них основные типы объектов, влияющих на скорость движения.
3. По данным Aster GDEM определим участки со сложным рельефом.
4. Построим карту стоимости движения: в каждой точке карты будет отражена стоимость прохода через эту точку.
5. Определим оптимальный маршрут между интересующими нас точками.
6. Экспортируем полученный маршрут.

Создание набора данных: теория

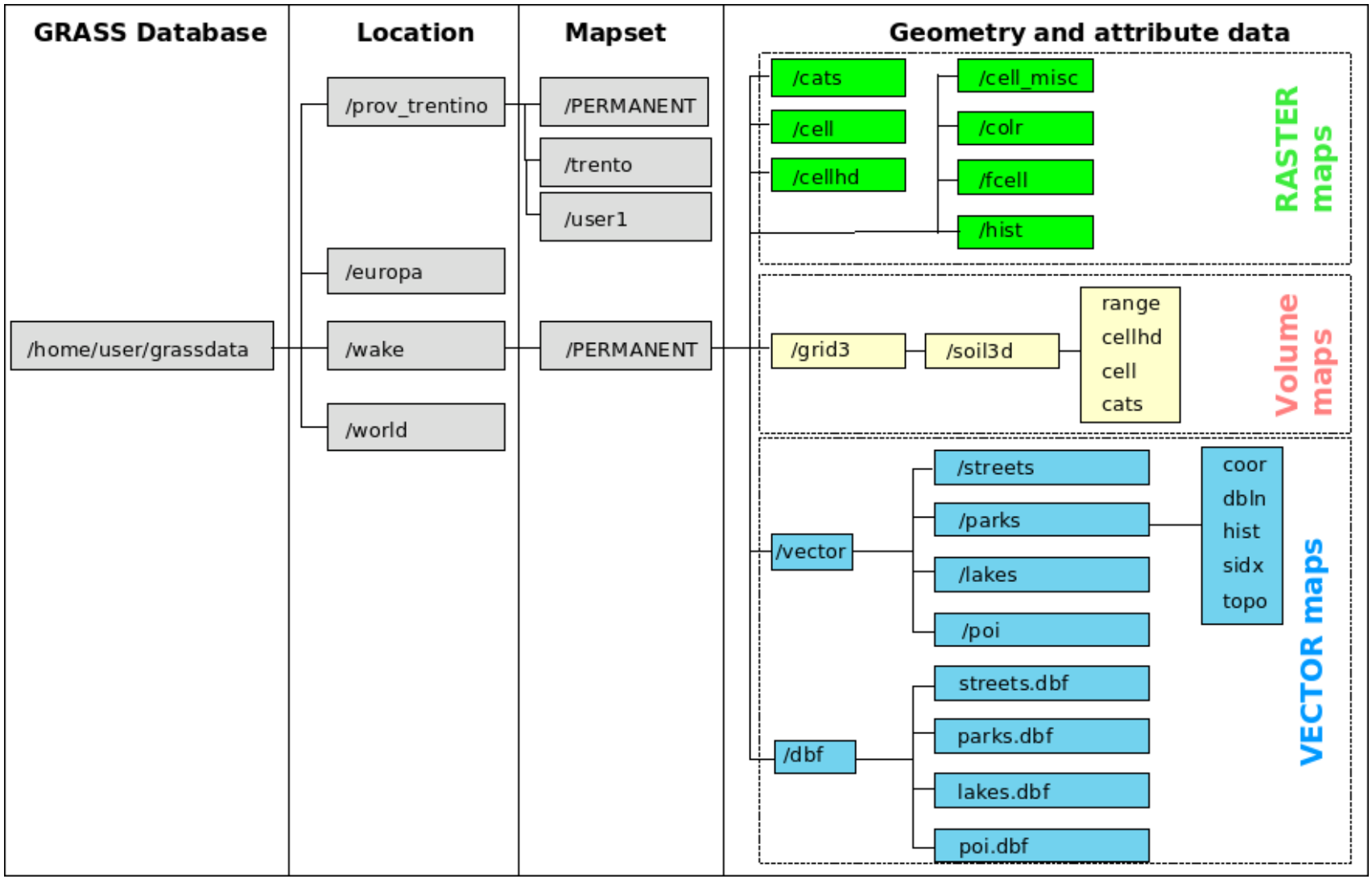
DATABASE — каталог данных. Здесь хранятся все проекты.
Аналогия: «здание библиотеки».

LOCATION — проект (область проекта). Проект определяется системой координат и охватом. В проекте хранится вся информация об интересующей территории (карты), настройки подключений к БД и т.п.
Аналогия: «читальный зал по конкретной отрасли знаний».

MAPSET — набор карт (набор данных). Часть проекта. Здесь могут храниться специфические карты (слои), для набора можно настроить отдельные права доступа для определенных пользователей, задать охват, отличный от охвата по умолчанию.
Аналогия: «книжный шкаф с книгами по определенной тематики или определенного автора».

REGION — вычислительный регион (охват). Текущее «окно», в котором выполняются все действия пользователя.

Типовая структура каталога данных



Запуск GRASS

Будем пользоваться интерфейсом Tcl/Tk. Для этого откроем терминал и запустим GRASS при помощи команды.

Команда запуска GRASS с интерфейсом TclTk из терминала

```
grass -tcltk
```

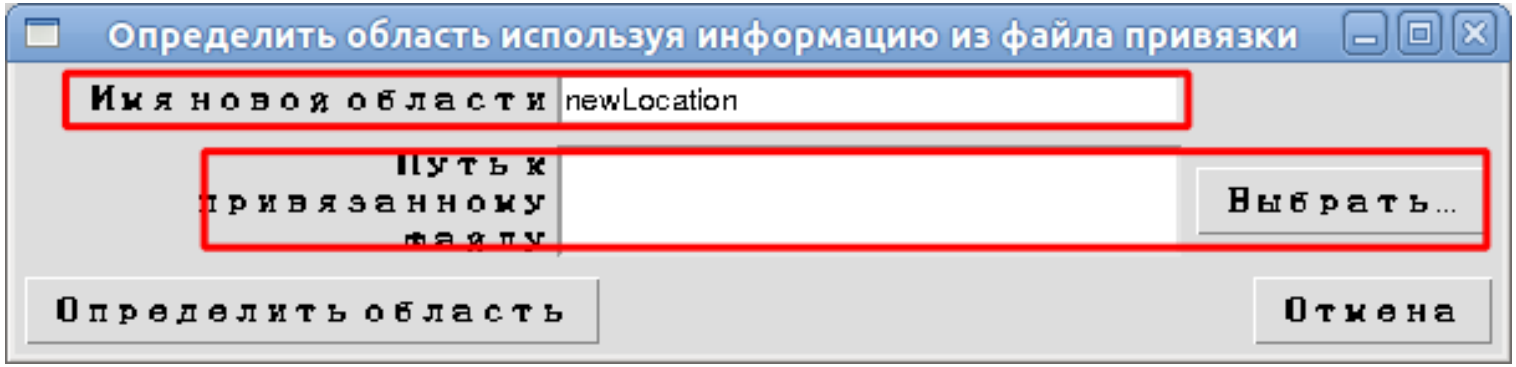
Создание набора данных

1. Вводим путь/название каталога ГИС-данных.
2. Выбираем «Define new location with . . . » -> «Georeferensed file».

Особенности:

- Пути к файлам не должны содержать пробелов.
- Названия должны быть заданы латиницей.

Создание набора данных

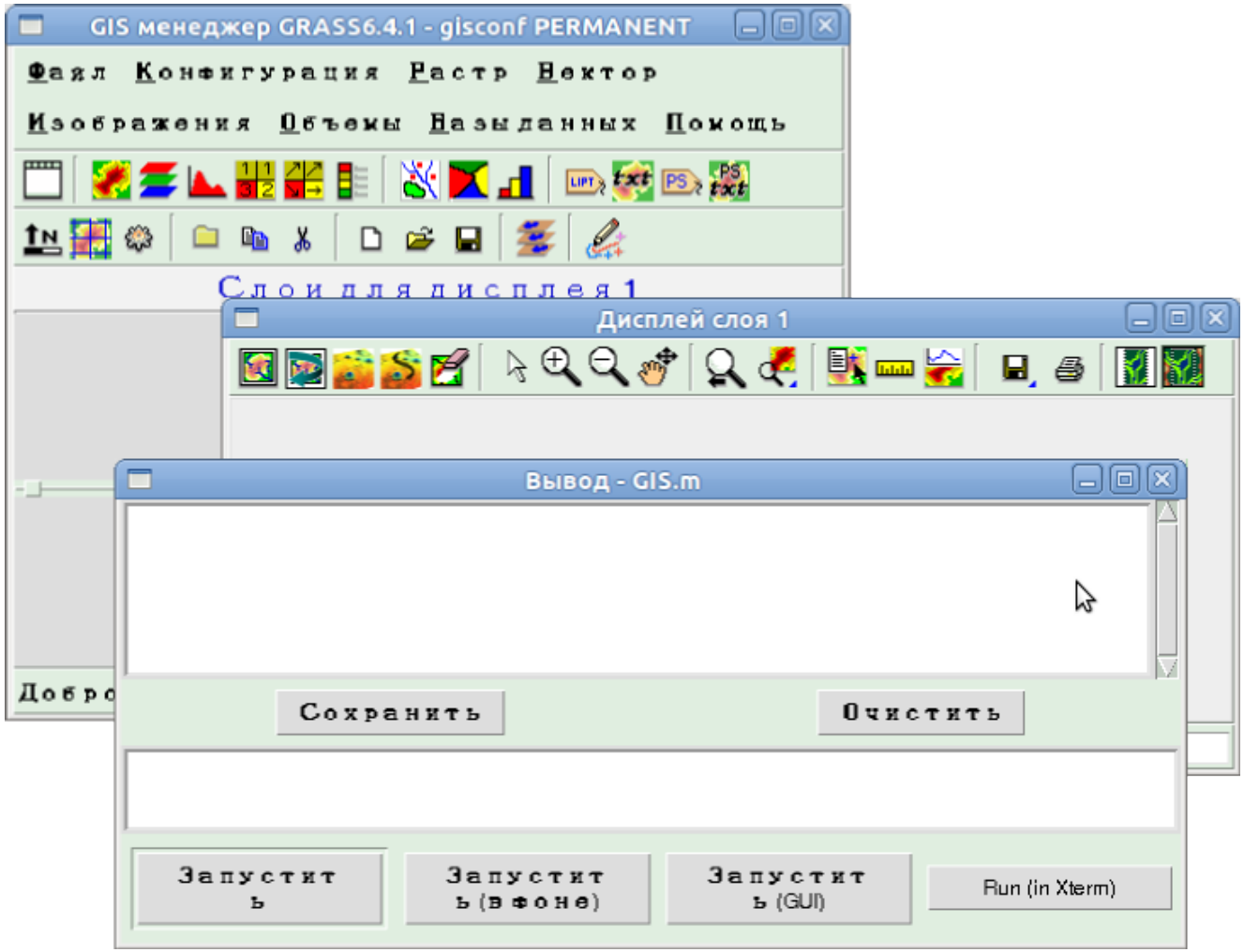


1. Задаем название области (например, gisconf).
2. Выбираем файл . . . /Landsat/L5_2010_07_23_B10.tif
3. Выбираем «Define location».

В стартовом окне появляется новая область «gisconf» и набор данных PERMANENT.

1. Выбираем область «gisconf».
2. Выбираем набор данных PERMANENT.
3. Выбираем «Enter GRASS».

Интерфейс Tcl/Tk: три основные окна и их назначение



Окно карты, окно ввода/вывода, GIS-менеджер.

Работа с текущим охватом (region)

ВАЖНО

Почти все действия над растрами происходят внутри виртуального прямоугольника — охвата. Для охвата можно настроить:

- границы (верхнюю/нижнюю, правую/левую);
- разрешение — размер ячейки растра (по горизонтали и вертикали).

Listing 1: Основные приемы использования команды

```
g.region -p
g.region res=RES t=TOP b=BOTTOM e=EAST w=WEST
```

Или из меню: «Config» -> «Region».

Вывод текущего охвата

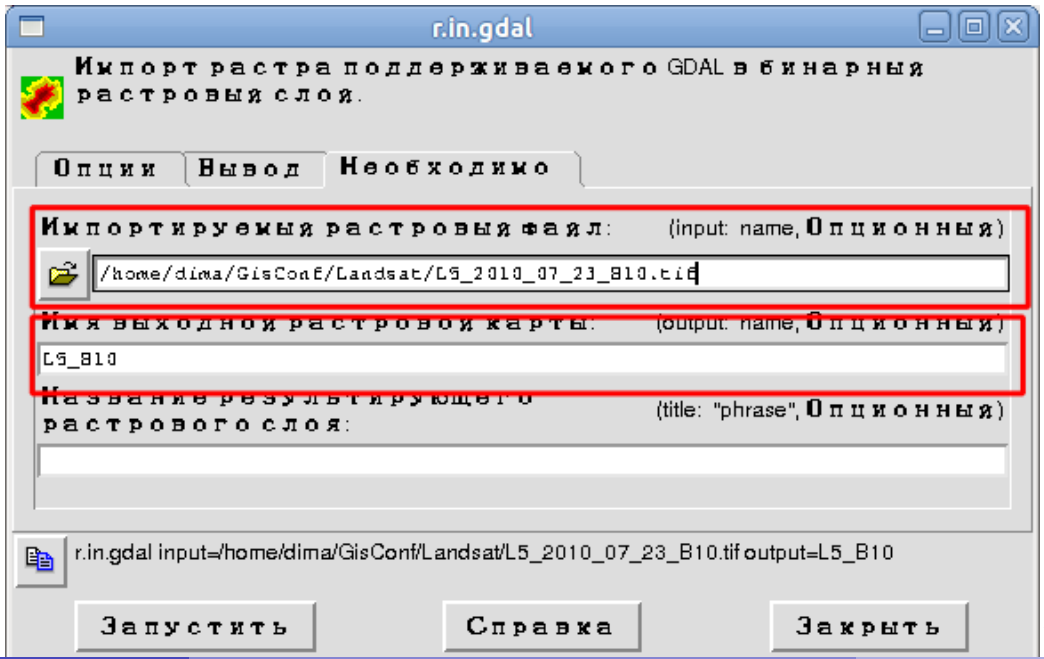
Listing 2: Параметры текущей области

```
> g.region -p
projection: 1 (UTM)
zone:       39
datum:      wgs84
ellipsoid:  wgs84
north:      6234825
south:      6225705
west:       357795
east:       368865
nsres:      30
ewres:      30
rows:       304
cols:       369
cells:      112176
```

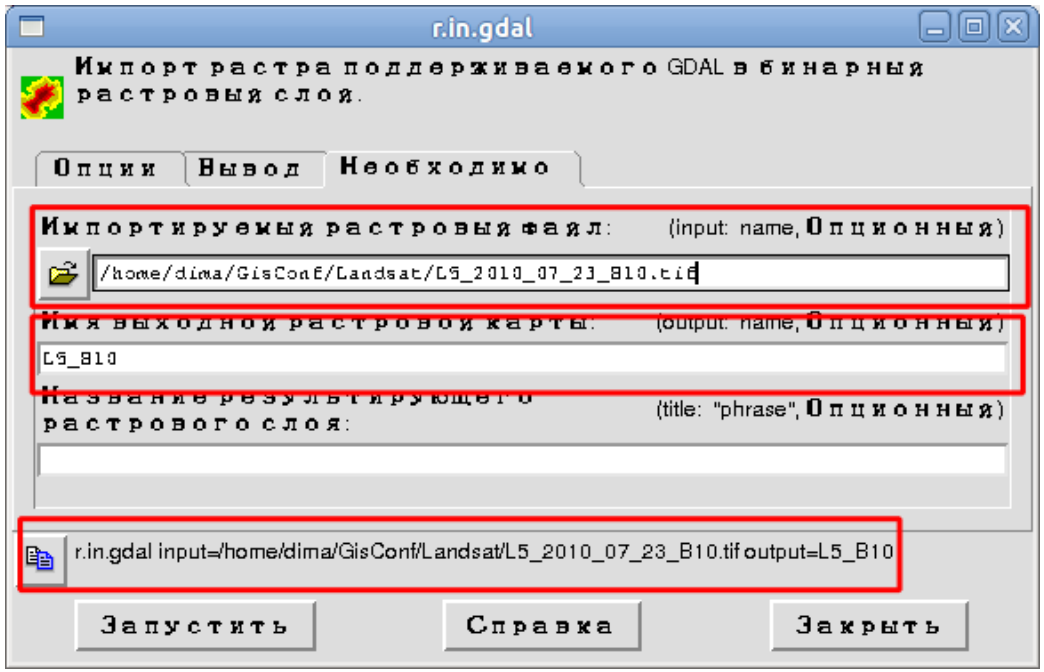
Импорт растровых данных

Импортируем данные Landsat:

1. Выбираем меню «File» -> «Import raster map» -> «Multiple formats, using GDAL».
2. В появившемся окне «r.in.gdal» выбираем вкладку «Required».
3. Выбираем импортируемый растровый файл: L5_2010_07_23_B10.tif.
4. Задаем название выходной растровой карты: L5_B10.



Импорт растровых данных: продолжение



Listing 3: Циклическая обработка

```
for NUM in B20 B30 B40 B50 B60 B70
do
   r.in.gdal in=~/GisConf/Landsat/L5_2010_07_23_$NUM.tif out=L5_$NUM
done
```

Импорт растровых данных: r.in.wms

«File» -> «Import raster map» -> «Web Map Server».
Необходимо указать:

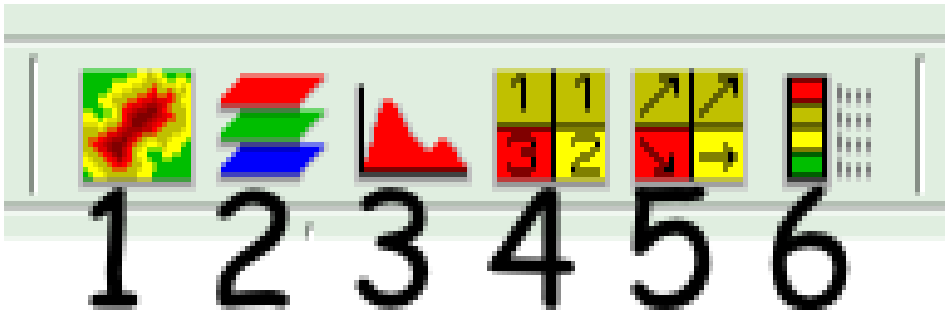
- Адрес сервера.
- Запрашиваемые слои.
- Исходная проекция (при несовпадении проекций GRASS произведет автоматическое перепроецирование).
- Метод интерполяции при перепроецировании.
- Формат запрашиваемого изображения.

Listing 4: Команда импорта SRTM / ASTER GDEM

```
r.in.wms maps=http://www.webservice-energy.org/mapserv/srtm \
    layers=srtm_s0 out=dem format=png
r.in.wms maps=http://wms.jpl.nasa.gov/wms.cgi layers=gdem \
    out=dem format=png
```

Визуализация растровых данных и получение статистики

Основные инструменты визуализации растровых данных:



Инструменты получения статистики и отчтетов («Raster» -> «Reports and statistics»):

- r.info
- r.report
- r.quantile
- ...

Импорт векторных данных

Импортируем данные OSM по республике Марий Эл (формат shp: `http://gis-lab.info/projects/osm_shp/region`).

Обратите внимание!

Данные имеют иную систему координат (широта/долгота, WGS-84), чем наш проект (UTM-39N).

Импортируем в новый проект RME_lonlat, который будет использовать систему координат импортируемых векторных слоев. Для этого: «File» -> «Import vector map» -> «Multiple formats, using OGR». Далее на вкладке «Options» в строку «Name for new location to create» вводим RME_lonlat. На вкладке «Required» выбираем имя импортируемого shp-файла highway-line.shp.
Или из командной строки:

```
v.in.ogr dsn=/home/dima/GisConf/shp/highway-line.shp \
        output=highway location=RME_lonlat
```

Переключение в новый проект и импорт данных

- Выбираем: «Config» -> «GRASS working environment» -> «Change working environment».
- Вносим название только что созданного проекта, в который хотим переключиться (RME_lonlat) и название набора (PERMANENT).
- Импортируем векторные данные в этот проект:

```
for file in /home/dima/GisConf/shp/*shp
do
  newname=$(basename $file | cut -d- -f1)
  v.in.ogr dsn=$file output=$newname
done
```

Перепроецирование данных в GRASS

- Переключаемся обратно в исходный проект.
- Перепроецируем векторные карты из RME_lonlat в рабочий проект:
  - Выбираем: «Vector» -> «Develop map» -> «Reproject vector». Затем на вкладке «Опции» указываем область RME_lonlat, а на вкладке «Source» указываем исходный векторный слой, например, highway.
  - Или из командной строки:

```
for map in highway landuse settlement vegetation water
do
  v.proj input=$map location=RME_lonlat
done
```

Обрезка векторных карт полигоном

Данные выходят за пределы нашей области интересов. Обрежем их:

1. Создадим полигон, содержащий границы области («Vector» -> «Generate area for current region»), назовем его study_region.

```
v.in.region study_region
```

2. Обрежем векторные карты («Vector» -> «Overlay maps» -> «Overlay»):

```
for map in highway water
do
  v.overlay ainput=$map atype=line binput=study_region \
    btype=area output=${map}_crop operator=and
done
for map in landuse vegetation
do
  v.overlay ainput=$map atype=area binput=study_region \
    btype=area output=${map}_crop operator=and
done
```

3. Выберем точки settlement, попадающие в область интересов («Vector» -> «Query with another map»).

Удаление исходных карт и переименование

1. Удалим ненужные карты: «File» -> «Manage maps and volumes» -> «Delete».

```
g.mremove -f vect=highway,landuse,settlement,vegetation,water
```

2. Переименуем карты (уберем _crop из названия): «File» -> «Manage maps and volumes» -> «Rename».

```
for map in $(g.mlist vect pattern="*_crop")
do
  newname=$(basename $map _crop)
  g.rename vect=$map,$newname
done
```

Анализ данных Landsat: действия

Произведем контролируемую классификацию данных Landsat. Для этого:

1. Построим композитные снимки для визуального анализа.
2. Построим индекс NDVI.
3. Выберем участки для обучающего множества.
4. Произведем обучение и классификацию.

Композитные снимки

Создадим композитный снимок: «Raster» -> «Manage map colors» -> «Create RGB».

```
r.composite r=L5_B40 g=L5_B30 b=L5_B20 out=landsat432
```

Расчет NDVI

Определение NDVI

$$NDVI = \frac{NIR - RED}{NIR + RED}$$

Радиометрическая коррекция

При регистрации данных Landsat физические величины яркости каналов сжимаются в диапазон [1,255] (преобразование L -> DN). Исходные максимальные и минимальные значения яркостей приводятся в метаданных к снимку:

LMAX_BAND3 = 264.000 LMIN_BAND3 = −1.170
LMAX_BAND4 = 221.000 LMIN_BAND4 = −1.510

Обратное преобразование (DN->L) производится по формуле:

$$L = \frac{LMAX\_BAND - LMIN\_BAND}{255 - 1}(DN - 1) + LMIN\_BAND$$

Реализация коррекции и расчет NDVI

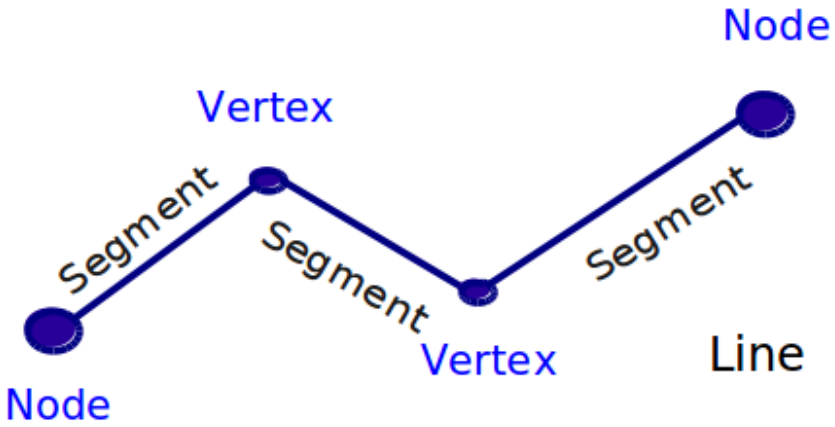
```
r.mapcalc "b3 = (264.0+1.17)*(L5_B30-1)/254 - 1.17"
r.info b3

r.mapcalc "b4 = (221.0 + 1.51)*(L5_B40)/254 -1.51"
r.info b4

r.mapcalc "ndvi = (b4 - b3)/(b4+b3)"
r.info ndvi

r.colors map=ndvi color=ndvi
```

Векторные данные

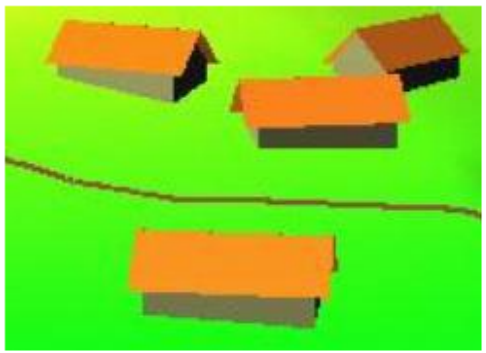
Point
Centroid
Line
Boundary
Area (boundary + centroid)
face (3D area)
[kernel (3D centroid)]
[volumes (faces + kernel)]

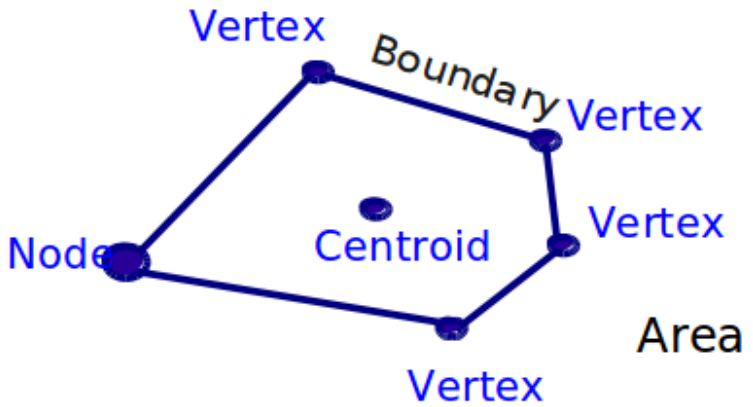
Геоданные **именно** 3D: x, y, z

Не во всех ГИС!



Faces

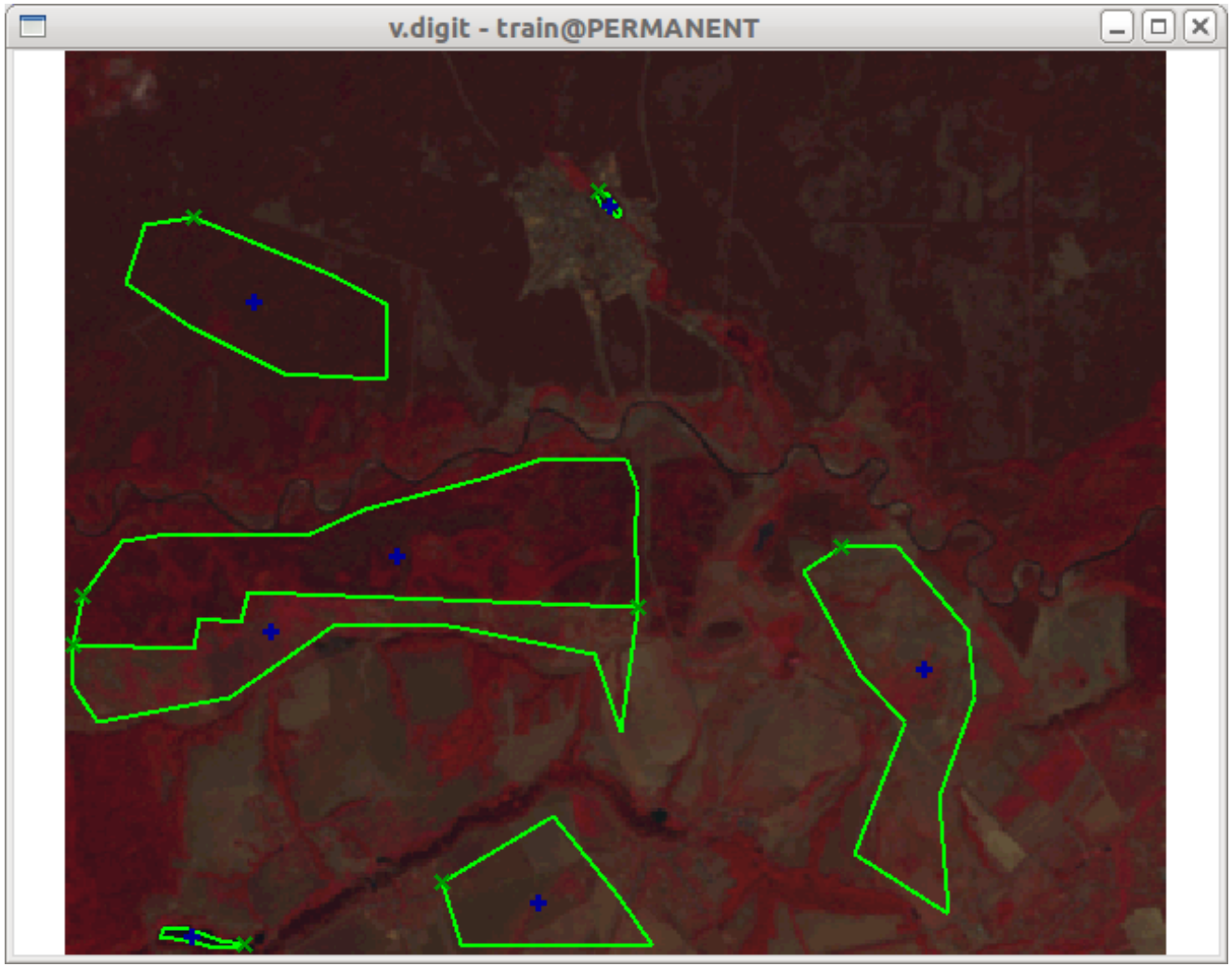




Построение обучающего множества для алгоритмов обучения

1. Выбираем «Vector» -> «Develop map» -> «Digitize»
2. Вводим имя результирующей векторной карты (train).
3. Команды отображения: d.rast landsat432.
4. Отметить «Create new file if it does not exist».
5. Запустить.
6. Открыть настройки, задать поля таблицы атрибутов (cat, description).
7. Отрисовать обучающие области.

Результат векторизации



Преобразование вектор -> растр

1. Выбираем «File» -> «Map type conversions» -> «Vector to raster».
2. Имя векторной карты — train.
3. Имя выходной растровой карты — train.
4. Использовать для значений растра категории вектора — cat.

или из командной строки:

```
v.to.rast input=train use=cat out=train
```

Отчет: зональная статистика

```
r.report -n train,ndvi un=p nsteps=4
r.report -n ndvi,train un=p nsteps=4
```

Создание группы изображений

1. Создадим группу изображений «Imagery» -> «Develop images and groups» -> «Create/edit group».
2. Группа «Landsat», подгруппа «all».
3. Растры:
   L5_B10,L5_B20,L5_B30,L5_B30,L5_B40,L5_B50,L5_B60,L5_B70

Или из командной строки:

```
i.group group=landsat subgroup=all \
    input=$(g.mlist rast pat="L*" sep=",")
```

Классификация методом максимального правдоподобия

Классификация происходит в два этапа:

- Извлечение статистических характеристик из обучающего набора данных (карта train) и настройка параметров модели («Imagery» -> «Classify image» -> «Input for supervised MLP»). Указываем карту содержащую классифицированные образцы (train), группу (landsat), подгруппу (all) снимков и название выходного файла со статистиками (all).

```
i.gensig trainingmap=train group=landsat subgroup=all \
    signaturefile=all
```

- Классификация снимков на основе полученных параметров. («Imagery» -> «Classify image» -> «Maximum likelihood classification MLP»). Указываем руппу (landsat), подгруппу (all) снимков, название файла со статистиками (all) и название карты, содержащей результаты классификации (lands).

```
i.maxlik group=landsat@PERMANENT subgroup=all \
    sigfile=all class=lands
```

Визуальный анализ рельефа

1. Для визуального анализа: «File» -> «3D-rendering» -> «NVIZ».
2. Выбираем Raster: название растра высот (dem) и растр, накладываемый на растр высот (ndvi). Вектор: векторная карта, накладываемая на растр высот (water).
3. Запустить, настроить параметры просмотра.

Анализ уклонов и экспозиций

1. Меняем разрешение региона под разрешение карты высот (dem): «Config» -> «Region»-> «Change region settings». Растр: dem
2. Выбираем «Raster» -> «Terrain analysis» -> «Slope and aspect».
3. Название растра высот (dem).
4. Названия выходных растров уклонов (slope) и экспозиции (aspect).

```
r.slope.aspect elevation=dem slope=slope aspect=aspect
```

Обзор

1. Построим растр стоимости, исходя из предположительной скорости движения через ячейку растра:
   - Дорога: 5 км/ч. Стоимость: 1/5.
   - Пастбище: 4 км/ч. Стоимость: 1/4.
   - Лес: 3 км/ч. Стоимость: 1/3.
   - Вода: 0.5 км/ч. Стоимость: 1/(0.5).
   - Склон (>10 градусов) замедляет движение в два раза.
2. Зададим начальную и конечную точку движения.
3. Найдем путь наименьшей стоимости между ними.

Растр стоимости

Рассчитаем растр стоимости.

1. Растеризуем карту дорог: Выбираем «File» -> «Map type conversions» -> «Vector to raster». (Имя растра: roads, имя векторной карты: highway).

```
v.to.rast highway out=roads use=val val=1
```

2. Растеризуем карту рек:

```
v.to.rast water out=water use=val val=1
```

3. Рассчитаем растр стоимости:

```
r.mapcalc "cost = 10"
r.mapcalc "cost = if(lands == 1, 0.333, cost)"
r.mapcalc "cost = if(lands == 2, 0.25, cost)"
r.mapcalc "cost = if(lands == 3, 2, cost)"
r.mapcalc "cost = if(isnull(water), cost, 2)"
r.mapcalc "cost = if(isnull(roads), cost, 0.2)"
r.mapcalc "cost = if(slope>10, cost*2, cost)"
```

Векторизуем точки

Векторизуем интересующие нас точки — озера. Сохраним в векторный слой с названием poi.

Построим растры зависимости стоимости от расстояния от точек

1. Определим координаты точек poi: «File» -> «Export vector map» -> «ASCII points or GRASS ASCII vector»

```
v.out.ascii input=poi
```

2. Построим растр стоимости движения к/от первой точки: «Raster» -> «Terrain analysys» -> «Cost surface». Имя выходного растра стоимости: p1, начальная точка (start): координаты первой точки интереса (363208.01009116,6233222.93525224).

```
r.cost -k in=cost out=p1 coord=363208.01009116,6233222.93525224
```

Поиск кратчайшего расстояния от одной точки до другой

Найдем кратчайшее расстояние от третьей точки до первой: «Raster» -> «Terrain analysys» -> «Least cost route or flow».
Выбираем растр, по которому будем искать путь меньшей стоимости (p1), название выходного растра — искомый путь (r31) и точку затравки (третья точка интереса с координатами 359215.01022308,6225993.90231572.

```
r.drain in=p1 out=r31 coord=359215.01022308,6225993.90231572
```

Можно найти расстояния от всех точек интереса, если указать название карты poi:

```
r.drain input=p1 output=r1 \
    vector_points=poi --overwrite
```

Цикл по всем точкам

Зациклим поиск кратчайших расстояний:

```
i=1
for coords in 363208.01009116,6233222.93525224 \
364740.86786043,6229892.28133384 \
359215.01022308,6225993.90231572 \
360880.33718228,6226429.15822551 \
361031.73054221,6226694.09660538
do
   r.cost -k in=cost out=p$i coord=$coords --o
   r.drain input=p$i output=r$i vector_points=poi --o
   i=$(expr $i + 1)
done
```

Затем проверим, что все было рассчитано верно, просматривая полученные карты.

Задача коммивояжера

Получили 5 карт расстояний. Нужно найти кратчайший путь между всеми 5-ю точками. Для этого:

1. Склеим карты в одну.
2. Векторизуем полученную карту.
3. Создадим дорожную сеть.
4. Найдем решение задачи.

Склейка карт

Склеиваем карты: «Raster» -> «Overlay» -> «Patch maps».
(Названия склеиваемых карт, название выходной карты).

```
r.patch input=r1,r2,r3,r4,r5 output=path
```

Автоматическая векторизация найденных маршрутов

Векторизуем найденные маршруты: «File» -> «Map type conversion» -> «Raster to vector». Имя растровой карты: path, имя получаемой векторной карты: path, тип объектов: line.

```
r.to.vect input=path output=path feature=line
```

Возможное сообщение об ошибке

Raster map is not thinned properly. Please run r.thin.
В этом случае нужно запустить модуль для предобработки «толстых» линий: «Raster» -> «Transform features» -> «Thin».

```
r.thin input=path output=path1
```

и повторить векторизацию для упрощенной карты.

Построение сети

Выбираем «Vector» -> «Network analysis» -> «Network maintenence». Название векторной карты: path; название точечного слоя: poi; название выходной векторной карты: net. Операция: connect; порог: 40.

```
v.net input=path points=poi \
    output=net operation=connect thresh=40
v.category net op=report
```

Задача коммивояжера

Выбираем «Vector» -> «Network analysis» -> «Traveling salesman analysys». Название векторной карты: net; название выходной вектороной карты: result. Номера категорий: 1-5.

```
v.net.salesman input=net output=result alayer=1 nlayer=2 ccats=1-5
```

Вывод результатов

Для экспорта воспользуемся: «File» -> «Export vector map» -> . . . .

Заключение

GRASS представляет собой очень мощный набор инструментов. Какую бы вы задачу не решали, если вы знаете, что хотите сделать, вы найдете способ сделать это при помощи GRASS.
Изучение GRASS требует определенного времени, но время, потраченное на знакомство с системой, быстро окупается.