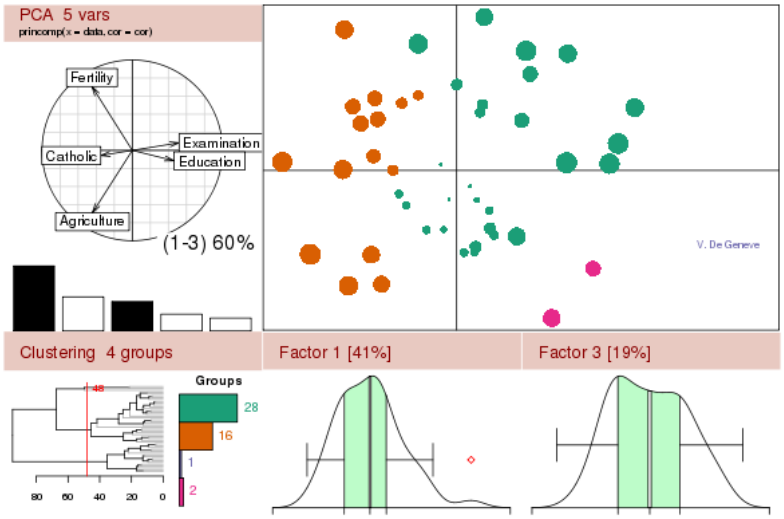


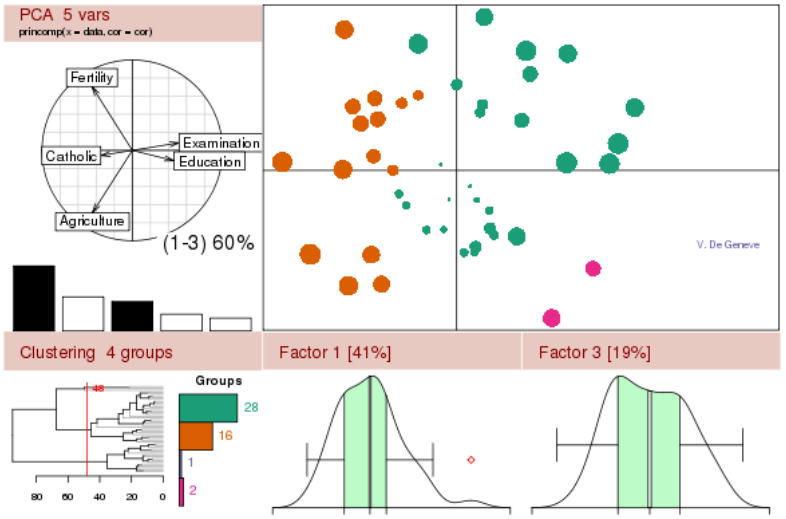
INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

R-Project para Computação Estatística e Gráficos

IFPB *Campus* Campina Grande

Parte 1

Aleciano Ferreira Lobo Júnior
aleciano@gmail.com



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

Slides cedidos pela Prof. Ana Cristina Alves de Oliveira Dantas que ministrou esse curso no ano de 2012.

Roteiro

- O que é o R-Project?
- Como obter?
- Recursos para Computação Estatística – Parte 1
- Prática!

3

O que é o R-Project?

- R é um ambiente para análises estatísticas e geração de gráficos
- Software livre
- Multiplataforma: Unix, Windows e MacOS

O que é o R-Project?

- Conjunto integrado de funcionalidades de software para manipulação de dados, cálculos e exibição gráfica
- Inclui:
  - Manipulação e armazenamento eficaz de dados;
  - Conjunto de operadores para cálculos sobre vetores (*arrays*) e matrizes;
  - Grande coleção coerente e integrada de ferramentas para análise de dados;
  - Gráficos para análise de dados, tanto para visualização na tela ou salvamento em arquivos;
  - Linguagem de programação simples e eficaz, que inclui condicionais, *loops*, funções definidas pelo usuário, recursividade e manipulação de entrada e saída de dados.

Como Obter?

- Sítio Web:
  - http://www.r-project.org/
- Escolha seu espelho:
  - Exemplo: http://cran-r.c3sl.ufpr.br/
- Opções de instalação:
  - Baixe e instale o R
  - Código fonte para todas as plataformas
    - ✓ http://cran-r.c3sl.ufpr.br/src/base/
      - R-latest.tar.gz

Espelhos no Brasil

- Universidade Federal do Paraná
  - http://cran-r.c3sl.ufpr.br/
- Fundação Oswaldo Cruz, Rio de Janeiro
  - http://cran.fiocruz.br/
- Universidade de São Paulo, São Paulo
  - http://www.vps.fmvz.usp.br/CRAN/
- Universidade de São Paulo, Piracicaba
  - http://brieger.esalq.usp.br/CRAN/

Escolha seu Sistema Operacional

Versão atual: 2.15.2

- Windows
  - Escolha a arquitetura 32/64 bit (47 MB)
  - http://cran-r.c3sl.ufpr.br/bin/windows/base/R-2.15.2-win.exe
- Linux
  - Debian
  - Redhat
  - Suse
  - Ubuntu

Exemplo de Instalação no Ubuntu (1/3)

- Versões suportadas:
  - Quantal Quetzal (12.10)
  - Precise Pangolin (12.04; LTS)
  - Oneiric Ocelot (11.10)
  - Natty Nawwhal (11.04)
  - Lucid Lynx (10.04; LTS)
  - Hardy Heron (8.04; LTS)

Exemplo de Instalação no Ubuntu (2/3)

- Para obter os últimos pacotes do R, adicione uma entrada (abaixo) no arquivo /etc/apt/sources.list, dependendo do SO:
  - deb http://<my.favorite.cran.mirror>/bin/linux/ubuntu quantal/
  - deb http://<my.favorite.cran.mirror>/bin/linux/ubuntu precise/
  - deb http://<my.favorite.cran.mirror>/bin/linux/ubuntu oneiric/
  - deb http://<my.favorite.cran.mirror>/bin/linux/ubuntu natty/
  - deb http://<my.favorite.cran.mirror>/bin/linux/ubuntu lucid/
  - deb http://<my.favorite.cran.mirror>/bin/linux/ubuntu hardy/
- Altere a URL do espelho (*mirror*) por um espelho de sua preferência (leve em consideração a proximidade geográfica!)

Exemplo de Instalação no Ubuntu (3/3) (recomendado)

- Para atualizar os pacotes:
  - sudo apt-get update
- Para instalar o R:
  - sudo apt-get install r-base
- Usuários que precisam compilar pacotes fontes do R fonte [por exemplo, mantenedores de pacotes, ou pacotes de terceiros instalados com install.packages ()] também devem instalar o pacote r-base-dev:
  - sudo apt-get install r-base-dev

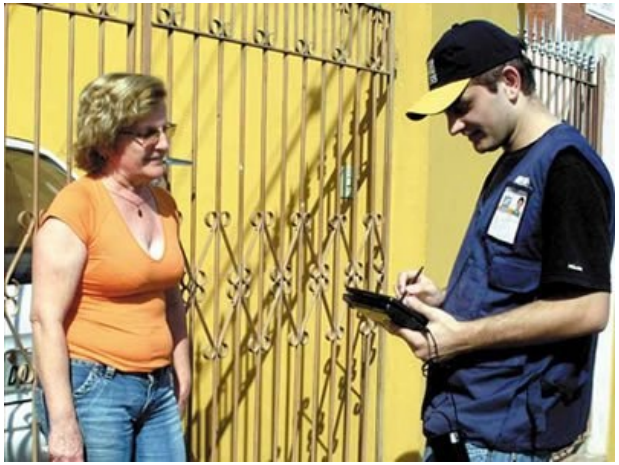
Exemplo de Compilação do Código-Fonte R-latest.tar.gz (1/2)

- Descompacte o arquivo:
  - tar –xzvf R-latest.tar.gaz
- Acesse o diretório da instalação (criado após a descompactação):
  - cd nome_do_diretorio
- Prepare o ambiente para a instalação:
  - ./configure
- Compile:
  - make
- Verifique se a compilação funcionou corretamente:
  - make check

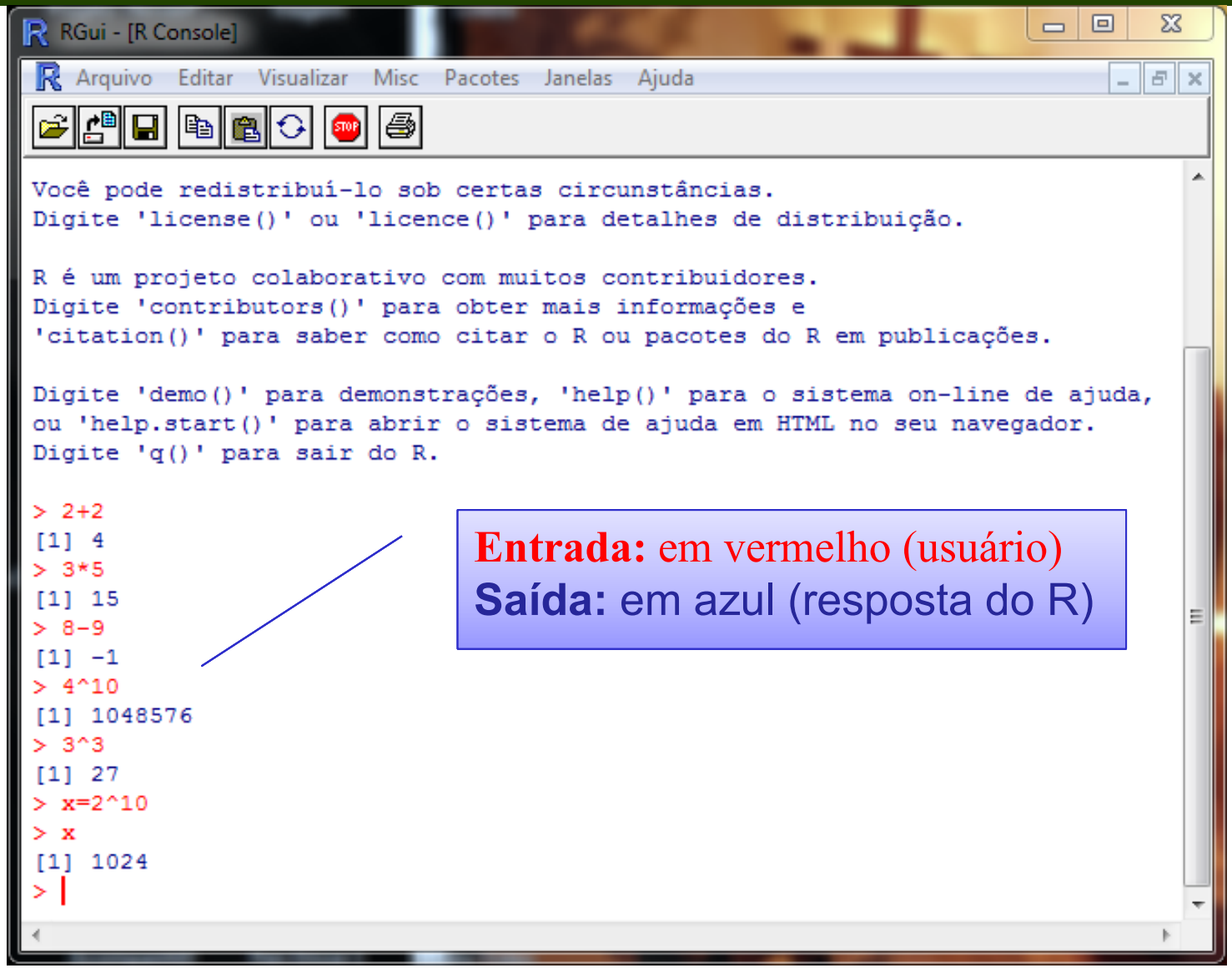
Exemplo de Compilação do Código-Fonte R-latest.tar.gz (2/2)

- Compile os manuais:
  - make pdf
  - make info
- Instale (requeridos privilégios de administrador):
  - make install
  - make install-info
  - make install-pdf
- Diretório padrão de instalação: /usr/local
  - Para alterar local de instalação, consultar o Manual de Instalação e Administração

O que o R pode nos oferecer para analisarmos nossos dados?



Operações Matemáticas



R como Calculadora

- O forma de uso mais básica do R é usá-lo como calculadora
- Os operadores matemáticos básicos são: + para soma, - subtração, * multiplicação, / divisão e ^ exponenciação
- Digite as seguintes operações na linha de comandos do R:
  - > **2+2**
  - > **2*3**
  - > **2/2**
  - > **2-2**
  - > **2^3**
- Use parênteses para separar partes dos cálculos, por exemplo, para fazer a conta 4+16, dividido por 4, elevado ao quadrado:
  - > **((4+16)/4)^2**
- Ignore o símbolo '>'. Está sendo apenas usado para indicar que você pode digitar seus comandos (**em negrito**)

Funções no R

- O R tem diversas funções que podemos usar para fazer os cálculos desejados
- O uso básico de uma função é escrever o nome da função e colocar os argumentos entre parênteses
  - Por exemplo: função(argumentos)
  - Função: especifica qual função irá usar
  - Argumentos: especifica os argumentos que serão avaliados pela função
- Com um pouco de prática eles se tornarão triviais!
- Vamos aprender como usar os arquivos de ajuda do R!

Usando Algumas Funções

- **Função sqrt():** é usada para calcular a raiz quadrada
  - > **sqrt(9)** # Tira a raiz quadrada dos argumentos entre parênteses, no caso 9
  - > **sqrt(3*3^2)** # raiz quadrada de 27
  - > **sqrt((3*3)^2)** # raiz quadrada de 81
- **Função prod(): é a função para multiplicação**
  - > **prod(2,2)** # O mesmo que 2x2
  - > **prod(2,2,3,4)** # 2x2x3x4
- **Função log(): é a função para calcular o logaritmo**
  - > **log(3)** # log natural de 3
  - > **log(3,10)** # log de 3 na base 10
  - > **log10(3)** # o mesmo que acima! log 3 na base 10
- **Função abs():** é a função para pegar os valores em módulo
  - > **abs(3-9)** # abs = modulo, |3-9|
- **Função factorial():** Para fazer o fatorial de algum número
  - > **factorial(4)** #4 fatorial (4!)

Ajuda no R

- Existem três formas básicas para descobrir uma função:
- Pesquisar entro do R usando palavras chave usando a função **help.search()**
  - **> help.search("Logarithms")**
- Nas versões mais recentes do R a função **help.search()** pode ser substituída por apenas **??** (duas interrogações)
  - **> ??logarithms**
- Buscar ajuda na internet, no site do R, com a função **RSiteSearch()**
  - **> RSiteSearch("logarithms")**# abre uma página na Internet
- Para ver os arquivos de ajuda do R, use o comando help(nome.da.função) ou ?nome.da.funcão
  - **> help(log) # abre o help sobre a função log**

Demonstrações do R

- Algumas funções do R possuem demonstrações de uso
  - Podem ser vistas usando a função **demo()**
- Vamos ver algumas demonstrações de gráficos que podem ser feitos no R!
- Digite a seguinte na linha de comandos:
  - > **demo(graphics)** # Vai aparecer uma mensagem pedindo que você tecle Enter para prosseguir, depois clique na janela do gráfico para ir passando os exemplos.
  - > **demo(persp)**
  - > **demo(image)**

Objetos no R

- Existem muitos tipos de objetos no R
- Por enquanto, vamos aprender os tipos básicos de objetos
  - **Vetores:** uma sequência de valores numéricos ou de caracteres (letras, palavras)
  - **Matrizes:** coleção de vetores em linhas e colunas, todos os vetores dever ser do mesmo tipo (numérico ou de caracteres)
  - **Dataframe:** O mesmo que uma matriz, mas aceita vetores de tipos diferentes (numérico e caracteres). Geralmente nós guardamos nossos dados em objetos do tipo *dataframe*, pois sempre temos variáveis numéricas e variáveis categóricas
  - **Listas:** conjunto de vetores, *dataframes* ou de matrizes. Não precisam ter o mesmo comprimento. É a forma que a maioria das funções retorna os resultados
  - **Funções:** funções criadas para fazer diversos cálculos também são objetos do R

Objetos vetores com valores numéricos

- Vamos criar um conjunto de dados de contem o número de alunos fazendo o minicurso de R em cada campus do IFPB na ST
- As quantidades de alunos são 22, 28, 37, 34, 13, 24, 39, 5, 33, 32:
  - > **alunos<-c(22,28,37,34,13,24,39,5,33,32)**
  - O comando **<-** (sinal de menor e sinal de menos) representa uma atribuição
  - A letra **c** significa **concatenar** (colocar junto)
- Para ver os valores (o conteúdo de um objeto):.
  - > **alunos**
- A função **length** fornece o número de observações:
  - > **length(alunos)**

Objetos vetores com caracteres (letras, variáveis categóricas)

- As letras ou palavras devem vir entre aspas " "
  - > **letras<-c("a","b","c","da","edw")**
  - > **letras**
  - > **palavras<-c("Manaus","Boa Vista","Belém","Brasília")**
  - > **palavras**
- Crie um objeto "misto", com letras e com números. Esses números realmente são números?
  - Note que na presença de aspas, os números são convertidos em caracteres
  - Evite criar vetores "mistos"

Operações com Vetores (1/2)

- **Podemos fazer diversas operações sobre vetores**
  - > **max(alunos)** #valor máximo contido no objeto alunos
  - > **min(alunos)** #valor mínimo
  - > **sum(alunos)** #Soma dos valores de alunos
  - > **alunos^2** #…
  - > **alunos/10**
- Vamos usar o que já sabemos para calcular a média dos dados das alunos!
  - > **n.alunos<-length(alunos)** # número de observações
  - > **media.alunos<-sum(alunos)/n.alunos** #média

Operações com Vetores (2/2)

- Para ver os resultados basta digitar o nome dos objetos que você salvou
  - > **n.alunos** # para ver o número de observações
  - > **media.alunos** # para ver a média
- O R já tem uma função pronta para calcular a média!
  - > **mean(alunos)**

Acessar Valores dentro de um Objeto (1/2)

- Caso queira acessar apenas um valor do conjunto de dados use colchetes “[]”
- Exemplos de valores nos objetos:
  - > **alunos[5]** # Qual o quinto valor de alunos?
  - > **palavras[3]** # Qual a terceira palavra?
- Para acessar mais de um valor use c para concatenar dentro dos colchetes [c(1,3,...)]:
  - > **alunos[c(5,8,10)]** # acessa o quinto, oitavo e décimo valores
- Para excluir um valor (o primeiro) use:
  - > **alunos[-1]** # note que o valor 22, o primeiro do objeto alunos, foi excluído

Acessar Valores dentro de um Objeto (2/2)

- Caso tenha digitado um valor errado e queira corrigir o valor, especifique a posição do valor e o novo valor:
- O primeiro valor de alunos é 22, caso estivesse errado e devesse ser 100, basta alterarmos o valor da seguinte maneira:
  - > **alunos[1]<-100** # O primeiro valor de alunos deve ser 100
  - > **alunos**
  - > **alunos[1]<-22** # Vamos voltar ao valor antigo
- Para acessar uma faixa de valores:
  - > **algunsAlunos<-alunos[3:10]** # Acessando do 3º ao 10º valor.

Transformar Dados

- Em alguns casos é necessário, ou recomendado, que você transforme seus dados antes de fazer suas análises
- Transformações comumente utilizadas são **log** e **raiz quadrada**
  - > **sqrt(alunos)** #Raiz quadrada dos valores de alunos
  - > **log10(alunos)** #log(alunos) na base 10, apenas
  - > **log(alunos)** # logaritmo natural de alunos
- Para salvar os dados transformados dê um nome ao resultado
  - > **alunos.log<-log10(alunos)** # salva um objeto com os valores de alunos em log

Listar e Remover os Objetos Salvos

- Para listar os objetos que já foram salvos use **ls()** que significa listar
  - > **ls()**
- Para remover objetos, use **rm()**
  - Remove o que está entre parênteses
  - > **rm(alunos)**
  - > **alunos** # você verá a mensagem:
    - ✓ *Error: object "alunos" not found*

Gerar Sequências

- Usar : (dois pontos) ou **seq()**
- **Dois pontos ":"** é usado para gerar seqüências de um em um, por exemplo a seqüência de 1 a 10:
  - > **1:10** # O comando : é usado para especificar seqüências.
  - > **5:16** # Aqui a seqüência vai de 5 a 16
- **Função seq():** usada para gerar seqüências especificando os intervalos
  - Vamos criar uma seqüência de 1 a 10 pegando valores de 2 em 2
  - > **seq(1,10,2)** #seq é a função para gerar seqüências, o default é em intervalos de 1.
  - **seq(from = 1, to = 10, by = 2)** # seqüência(de um, a dez, em intervalos de 2)
  - > **seq(1,100,5)** #seq. de 1 a 100 em intervalos de 5
  - > **seq(0.01,1,0.02)**

Gerar Repetições

- **Função rep():** para repetir algo n vezes
  - > **rep(5,10)** # repete o valor 5 dez vezes
  - > **rep(x, times=y)** # rep(repita x, y vezes) # onde *x* é o valor/conjunto que deve ser repetido, e *times* é o número de vezes)
  - > **rep(2,5)**
  - > **rep("a",5)** # repete a letra "a" 5 vezes
  - > **rep(1:4,2)** # repete a sequência de 1 a 4 duas vezes
  - > **rep(1:4,each=2)** # note a diferença ao usar o comando each=2
  - > **rep(c("A","B"),5)** # repete A e B cinco vezes.
  - > **rep(c("A","B"),each=5)** # repete A e B cinco vezes.
  - > **rep(c(“Cachorro”,“Gato”,“Peixe”),c(4,2,1))** # a primeira parte do comando indica as palavras que devem ser repetidas e a segunda parte indica quantas vezes cada palavra deve ser repetida

Gerar Dados Aleatórios (1/3)

- **runif (dados aleatórios com distribuição uniforme)**
  - > **runif(n, min=0, max=1)** # gera uma distribuição uniforme com n valores, começando em min e terminando em max.
  - > **runif(200,80,100)** # gera 200 valores que vão de um mínimo de 80 até um máximo de 100
  - > **temp<-runif(200,80,100)**
  - > **hist(temp)** # Faz um histograma de frequências dos valores
- **rnorm (dados aleatórios com distribuição normal)**
  - > **rnorm(n, mean=0, sd=1)** # gera n valores com distribuição uniforme, com média 0 e desvio padrão 1.
  - > **rnorm(200,0,1)** # 200 valores com média 0 e desvio padrão 1
  - > **temp2<-rnorm(200,8,10)** # 200 valores com média 8 e desvio padrão 10
  - > **hist(temp2)** # Faz um histograma de frequências dos valores

Gerar Dados Aleatórios (2/3)

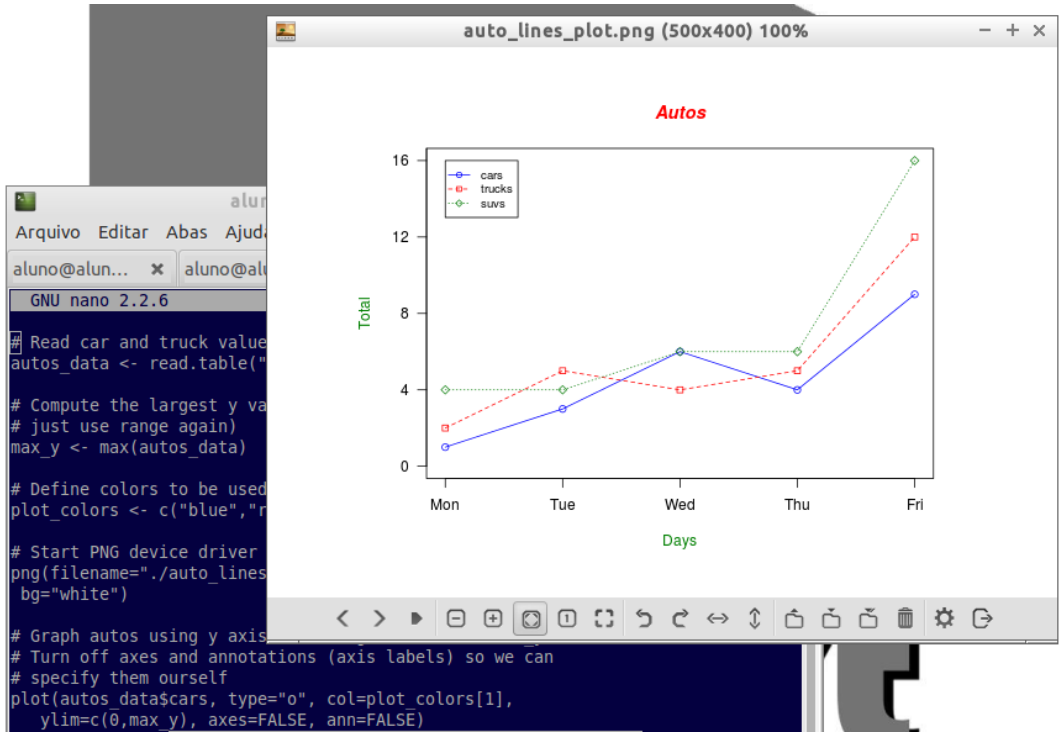
- Veja o help da função **?Distributions** para conhecer outras formar de gerar dados aleatórios com diferentes distribuições
- **Função sample():** fazer amostras aleatórias
  - **sample(x, size=1, replace = FALSE)** # onde *x* é o conjunto de dados do qual as amostras serão retiradas, *size* é o número de amostras e *replace* é onde você indica se a amostra deve ser feita com reposição (TRUE) ou sem reposição (FALSE)
  - > **sample(1:10,5)** # tira 5 amostras com valores entre 1 e 10
- Como não especificamos o argumento **replace**, o padrão é considerar que a amostra é sem reposição (= FALSE)
  - > **sample(1:10,15)** # erro, amostra maior que o conjunto de valores, temos 10 valores (1 a 10) portanto não é possível retirar 15 valores sem repetir nenhum!
  - > **sample(1:10,15,replace=TRUE)** # agora sim!

Gerar Dados Aleatórios (3/3)

- Com a função **sample** nós podemos criar vários processos de amostragem aleatória
- Por exemplo, vamos criar uma moeda e "jogá-la" para ver quantas caras e quantas coroas saem em 10 jogadas:
  - > **moeda<-c("CARA","COROA")** # primeiro criamos a moeda
  - > **sample(moeda,10)**
- Ops! Esqueci de colocar **replace=TRUE**:
  - > **sample(moeda,10,replace=TRUE)** # agora sim !

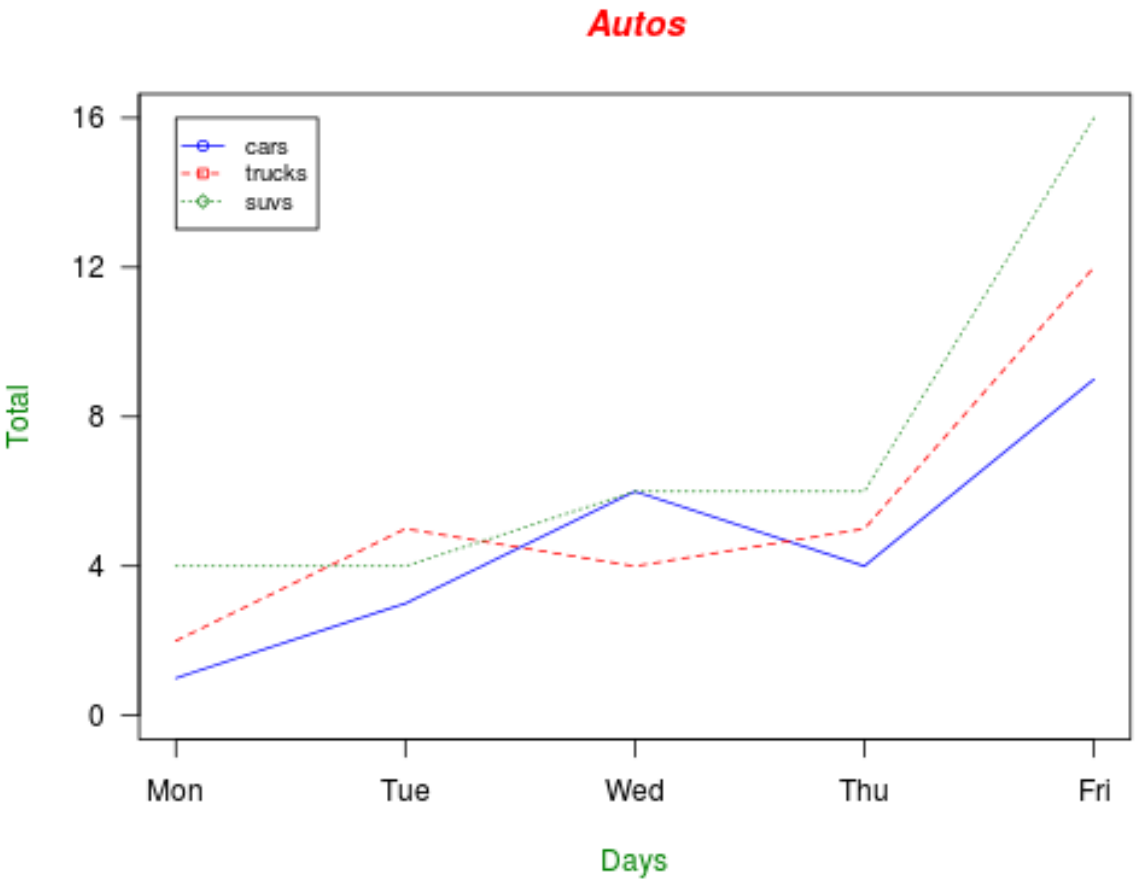
Gráficos!

- Os gráficos são um dos principais motivos pelo qual o R é buscado.
- **> demo(graphics)**



Gráficos!

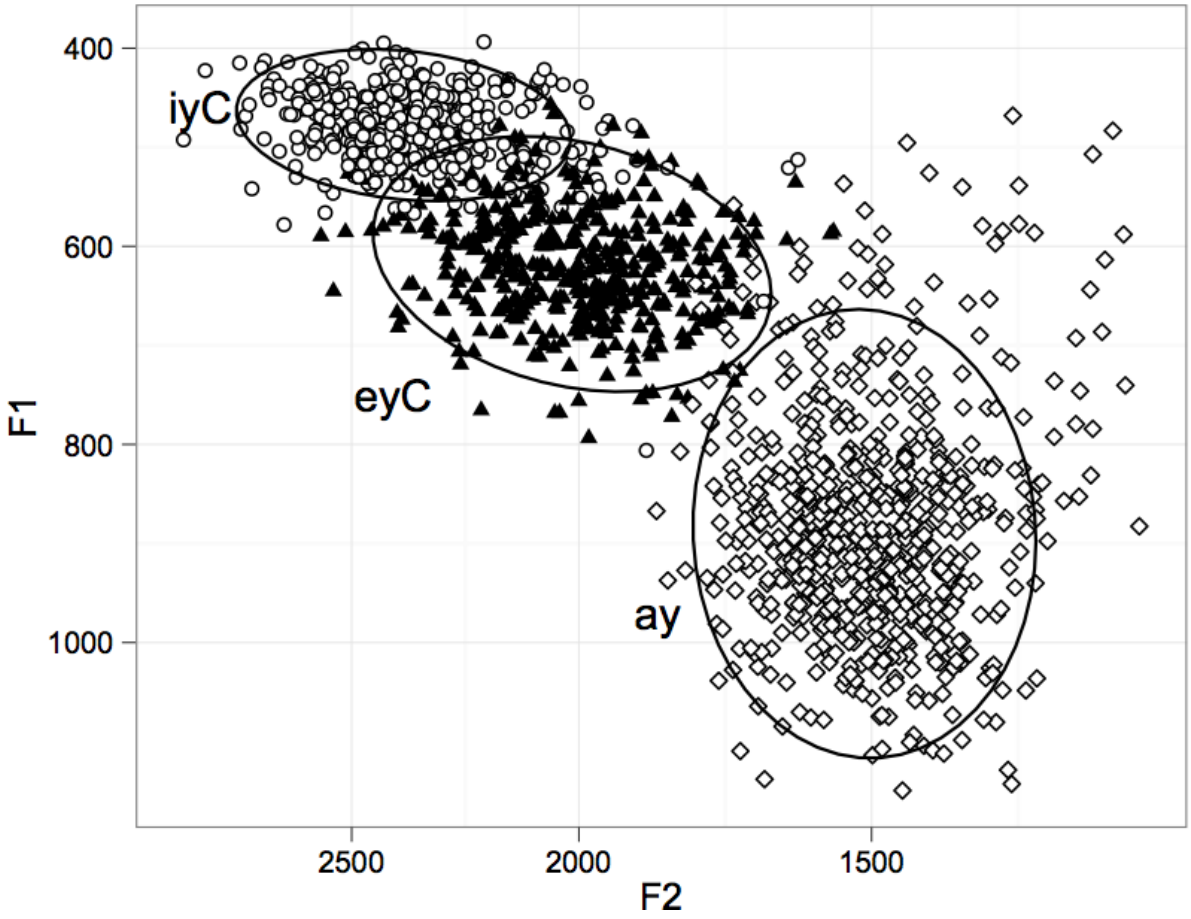
- Gráfico de linhas



> plot(..., type="l", ...)

Gráficos!

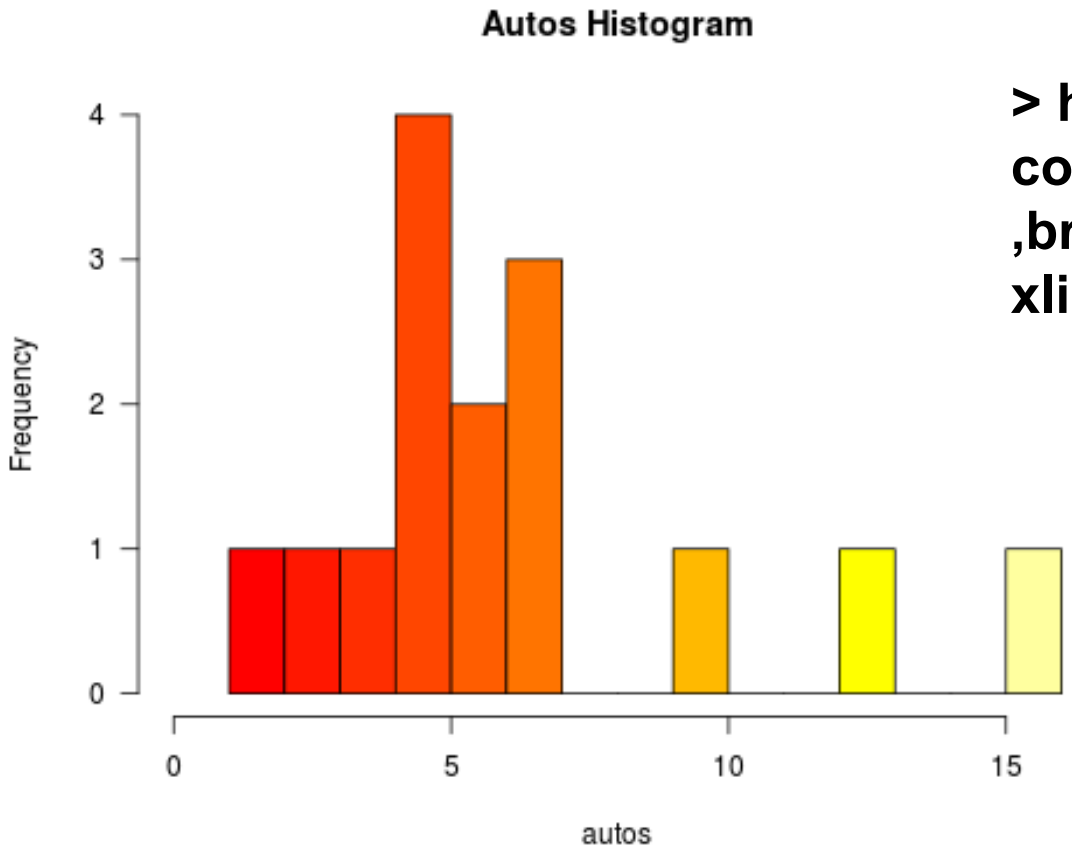
- Gráfico de pontos



> plot(..., pch=22, lty=2, ...)

Gráficos!

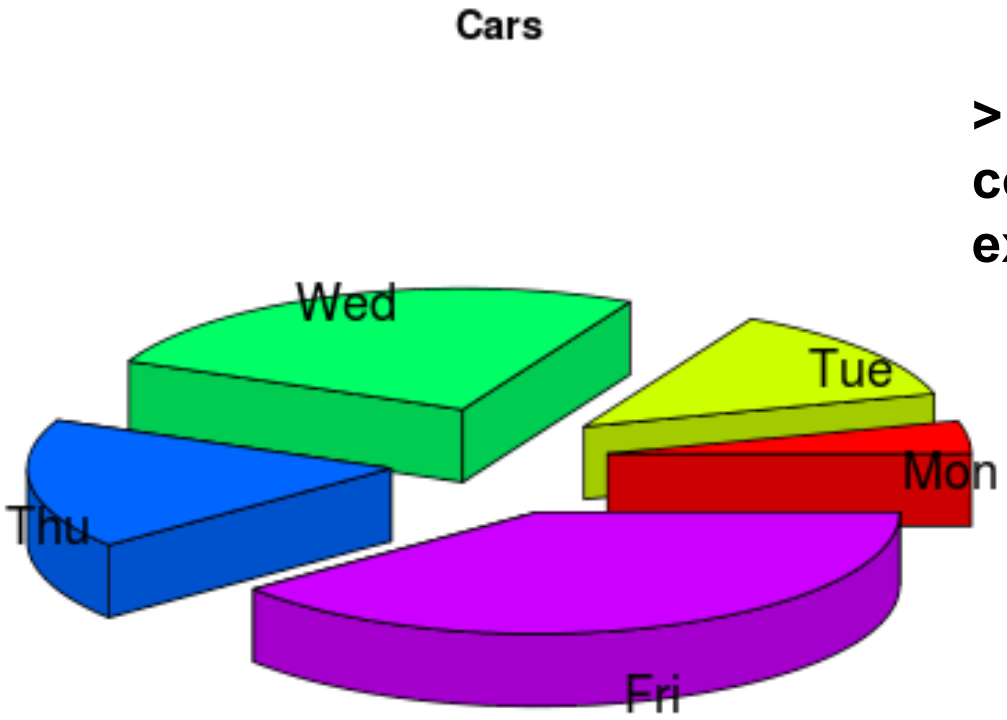
- Histograma



> hist(autos, ...,
col=heat.colors(max(autos))
,breaks=max(autos),
xlim=c(0,max(autos)), ...)

Gráficos!

- Gráfico de “pizza”



> pie(cars, ...,
col=rainbow(length(cars),
explode=0.3, ...)

Gráficos!

- Possibilidades:
  - Nomear eixos e título do gráfico;
  - Colocar texto em qualquer parte;
  - Utilizar conjunto de cores pré-caracterizado;
  - Colocar mais de um conjunto de dados no mesmo gráfico;
  - Salvar em arquivo para formatos **PNG**, **JPG**, **SVG** e etc.;

O que é o *Workspace*?

- Quando salvamos um *workspace* em uma pasta e o abrirmos a partir dela, o R imediatamente reconhecerá o endereço da pasta como sendo seu diretório de trabalho
- Para conferir o diretório de trabalho use o comando abaixo:
  - > **getwd()**
    - ✓ > [1] “/home/aluno"
- Quando você salva o *workspace*, as variáveis ficam guardadas com os valores que estão no momento do encerramento do R e podem ser manipuladas posteriormente

O que é o *Workspace*?

- Salve o seu *workspace* atual!
  - > **save(file="my.Rdata")**
- Carregue o seu *workspace*
  - > **load(file="my.Rdata")**

Scripts em R

- **Scripts:** Automatizar a criação de gráficos e procedimentos que são realizados repetidamente.
  - Acesse: ftp://192.168.1.212
  - User: aluno
  - Password: <vazio>
  - Acessar arquivo *script_simples.R* no diretório ***scripts*** do servidor FTP.
  - Estando no mesmo diretório que salvou o script:
  - **> R < ./script_simples.R --no-save**
  - O que aparece na tela?

Scripts em R

- **Ao rodar comandos do R através de scripts, os comandos para plotar gráficos não exibem nada pois não estão sendo executados de dentro do ambiente do R!**
- **Solução:** salvar em imagens. Para isso:
  - Adicionar no início do arquivo:
  - **png(filename="./script_simples.png",height=400, width=500)**
  - Adicione ao final do arquivo:
  - dev.off()

  - O que aparece na tela?

Coleta e Análise de Dados

- **Coleta:** utilizar algum script (programa na linguagem *shell script*) para pegar dados e salvar em algum arquivo de texto, por exemplo.
- **Análise:** utilizar o R para manipular estes dados e plotar gráficos.

Prática com coleta e análise

- Acesse: ftp://192.168.1.212
- User: aluno
- Password: <vazio>
- Acesse o diretório **scripts**
- Baixe os arquivos **plot_cpu_use.R e script_CPU.sh** para o diretório **home** de aluno
- Dê permissão de execução ao script principal:
  - **> sudo chmod 744 script_CPU.sh**
- **Execute-o:**
  - **> ./script_CPU.sh**

Exercícios parte 1

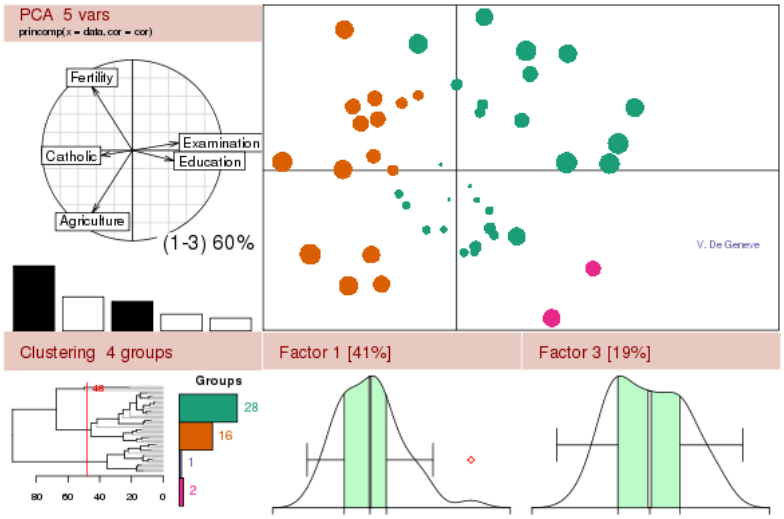
- Acesse: ftp://192.168.1.2
- User: aluno
- Password: <vazio>
- Acesse o diretório **exercicios**
- Baixe o arquivo **Minicurso-R-Exercicio01.pdf**

Referências

- Introdução ao uso do programa R. http://cran.r-project.org/doc/contrib/Landeiro-Introducao.pdf
- Tópicos de Estatística Utilizando R. http://cran.r-project.org/doc/contrib/Itano-descriptive-stats.pdf
- Amaral, Marcelo R. S.; Cesario, Carolina V. Apostila do Minicurso: Software R. http://www.ime.uerj.br/~mrubens/slae/minicursosoftwareR.pdf
- Manuais disponíveis na página do R: Exemplo "*An introduction to R*"

Referências

- Introdução ao Sistema Livre R em Análises Estatísticas. http://www.ecologia.ufrgs.br/~adrimelo/lm/apostilas/Mini_tutorial_R.pdf
- Estatística com R. Uma Iniciação para o Ensino Básico e Secundário. http://homepage.ufp.pt/cmanso/ALEA/dossie14_estat%A1stica%20com%20R.pdf
- R Data Import/Export (presente após instalação).



INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PARAÍBA

R-Project para Computação Estatística e Gráficos

IFPB *Campus* Campina Grande

Parte 1

Aleciano Ferreira Lobo Júnior
aleciano@gmail.com