



WORK VIII 2025 Eta

INSCREVA-SE JÁ!



WORKSHOP EM MODELAGEM NUMÉRICA DE TEMPO,
CLIMA E MUDANÇAS CLIMÁTICAS UTILIZANDO O MODELO ETA

25 a 29 de agosto

ONLINE

Introdução ao GrADS

VIII WorkEta 2025

Grid Analysis and Display System (GrADS)

É uma ferramenta interativa que é usada para acesso, manipulação, e visualização de dados científicos.

O GrADS trabalha com diversos formatos de dados:

- Binário
- Grib
- NetCDF
- Shapes files

Iniciando o Grads

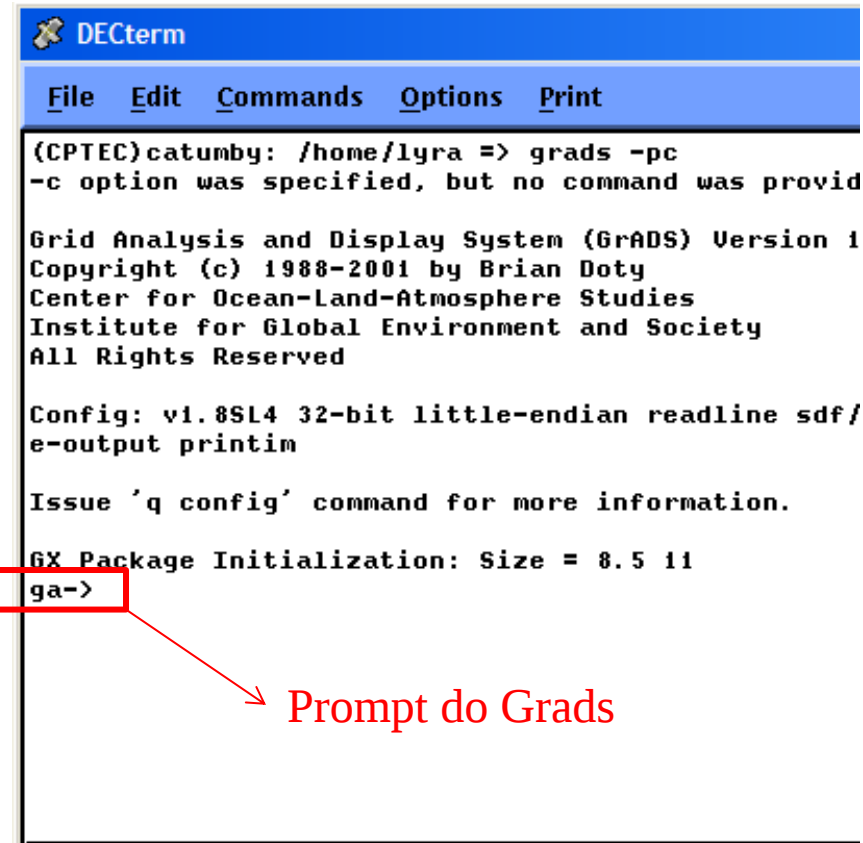
- Comando: `grads`
 - opções
 - b “batch mode” (sem janela gráfica)
 - l “landscape” (paisagem - 11 x 8,5 pol.)
 - p “portrait” (retrato - 8,5 x 11 pol.)
 - c executa o comando fornecido como o 1º comando após ser inicializado

- Exemplos:

`gradsc -p`

`gradsc -l`

`gradsc -pc “open Eta_template.ctl”`



```
(CPTEC)catumby: /home/lyra => grads -pc
-c option was specified, but no command was provided

Grid Analysis and Display System (GrADS) Version 1.0.0
Copyright (c) 1988-2001 by Brian Doty
Center for Ocean-Land-Atmosphere Studies
Institute for Global Environment and Society
All Rights Reserved

Config: v1.8SL4 32-bit little-endian readline sdf/
e-output printim

Issue 'q config' command for more information.

GX Package Initialization: Size = 8.5 11
ga->
```

Prompt do Grads

Arquivos descritores

- Contém informações sobre o conjunto de dados
- Extensão: **.ctl**
- Formato: **ASCII**

Exemplo:

```
DSET ^exemplo.bin
UNDEF 1e+20
XDEF 240 linear -106.00 0.40
YDEF 200 linear -53.00 0.40
ZDEF 7 levels 1000 850 700 500 300 200 100
TDEF 5 linear 00Z2jan1987 1dy
VARS 5
ps 0 99 Surface pressure [hPa]
ts 0 99 Surface (2m) air temperature [K]
p 0 99 Total precipitation rate [kg/(m^2*s)]
u 7 99 Eastward wind [m/s]
v 7 99 Northward wind [m/s]
ENDVARS
```

→ Nome do arq com conjunto de dados

→ Valor para dados ausentes

→ nº de pontos em x, longitude oeste, res

→ nº de pontos em y, latitude sul, res

→ nº de níveis, níveis

→ nº de tempos, tempo inicial, incremento

→ nº de variáveis



descrição das variáveis

Operações básicas

- Abrir arquivos

`open` [*arquivo descritor .ctl*]

Exemplo: ga-> `open Eta_40km.ctl`

`gradsc -pc "open Eta_40km.ctl"`

Scanning description file: curso_grads/Eta_40km.ctl

Data file curso_grads/Eta_40km.bin is open as file 1

LON set to -83 -25.8

LAT set to -50.2 12.2

LEV set to 1020 1020

Time values set: 2019:3:19:0 2019:3:19:0

E set to 1 1

Informações que aparecem na
abertura de um arquivo ctl

Operações básicas

- Listar as variáveis contidas no arquivo

`q file`

- Visualizar variáveis ou expressões

`d [variável]` Ex: `d tp2m`

`d [expressão]` → **Expressões:** `+` `*` `-` `/` **ou Função**

Ex: `d tp2m-273.15`

- Limpar

`clear` ou `c`

- Sair

`quit`

Operações básicas

- Limitar o domínio

set lat [*latitude*]

set lat [*latitude sul*] [*latitude norte*]

set lon [*longitude*]

set lon [*longitude oeste*] [*longitude leste*]

- Definir nível de pressão

set lev [*nível em hPa*]

- Definir instante de tempo

set t [*tempo*]

set t [*tempo1*] [*tempo2*]

set time [*hora*]Z[*dia*][*mês*][*ano*]

set time [*hora1*]Z[*dia1*][*mês1*][*ano1*] [*hora2*]Z[*dia2*][*mês2*][*ano2*]

Operações básicas

- Controlar o intervalo de contorno

`set cint [valor]`

- Controlar o valor mínimo de contorno

`set cmin [valor]`

- Controlar o valor máximo de contorno

`set cmax [valor]`

Janela gráfica

- Controlar a exibição do logotipo

`set grads` on/off

- Definir padrão ou cor da janela gráfica

`set display` [*mode*] [*color*]

[*mode*] = grey, greyscale, color [*color*] = white, black

*Sempre seguido de um comando `clear` para limpar a tela

Controlando ambiente de mapas

- Características das linhas de grade

`set grid [status] [estilo] [color]`

`[status]` = on, off, horizontal ou vertical

- Características do mapa

`set map [color] [estilo] [espessura]`

`[cor]` = (1-15)

`[espessura]` = (1-6)

`[estilo]` =

1 sólida	
2 traço largo	
3 traço curto	
4 traço longo traço curto	
5 ponto	
6 ponto traço	
7 ponto ponto traço	

Controlando ambiente de mapas

- Mudar o mapa padrão

`set mpdset [mapa_res]`

`[mapa_res]` = lowres, mres, hires, brmap_hires, **amsulrp**

- Mudar a projeção

`set mproj [projeção]`

`[projeção]` = latlon, scaled, nps, sps, lambert, ...

- Retirar ou colocar o mapa

`set mpdraw [on/off]`

Controle de página

- Página Virtual

`set vpage [xmin] [xmax] [ymin] [ymax]`

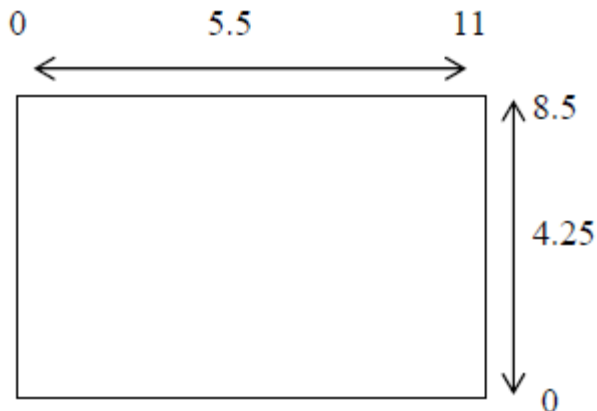
`set vpage off`

- Área de plotagem – Não é apropriado para múltiplos plots em uma página

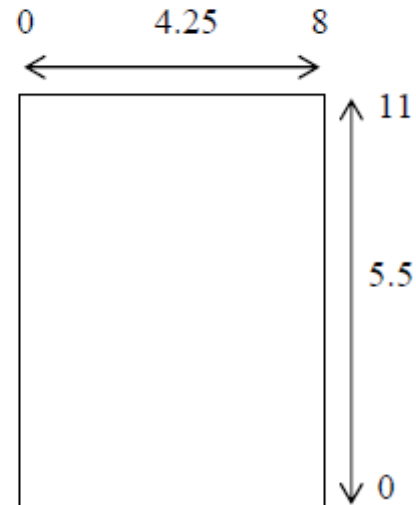
`set parea [xmin] [xmax] [ymin] [ymax]`

`set vpage off`

Tamanhos padrões da tela de visualização:
`grads -l` (landscape: 11 x 8.5)



`grads -p` (portrait: 8.5 x 11)



Controlando a orientação dos eixos

- Inverter os eixos

`set xyrev [on/off]`

- Vira a ordem do eixo x

`set xflip [on/off]`

- Vira a ordem do eixo Y

`set yflip [on/off]`

Controlando a rotulação dos eixos

- Início e fim do eixo

`set xaxis [inicial] [final] [incr]`

`set yaxis [inicial] [final] [incr]`

- Intervalo de rotulação do eixo

`set xlint [intervalo]`

`set ylint [intervalo]`

- Opções do eixo

`set xlopts [cor] [espessura] [tamanho]`

`set ylopts [cor] [espessura] [tamanho]`

Variáveis

- Variáveis pré-definidas

lat, lon, lev

- Definir uma variável

`define [var] = [expr]`

`[var]` = nome da variável

`[expr]` = expressão, função matemática

Pode ser usada em um comando subsequente `define`, `display` ou `d`

Ex: `define zave=ave(temp,t=1,t=30)`

`undefine [var]`

Funções matemáticas

- Média

ave (*expr*, *dim1*, *dim2*, *<tinc>*, *<-b>*)

expr - expressão

dim1 - ponto inicial (Ex: t=1)

dim2 - ponto final (Ex: t=12)

tinc - incremento

-b - contorno exato

Ex: **ave**(tp2m,t=1,t=120,4)

Média na área

aave (*expr*, *xdim1*, *xdim2*, *ydim1*, *ydim2*)

expr - expressão

xdim1 - dimensão mais a oeste (Ex: lon=0 ou x=1)

xdim2 - dimensão mais a leste (Ex: lon=360 ou x=180)

ydim1 - dimensão mais a sul (Ex: lat=-90 ou y=1)

ydim2 - dimensão mais a norte (Ex: lat=90 ou y=90)

Ex: **aave**(tp2m,x=1,x=72,y=1,y=46)

Funções matemáticas

- Somatório

sum (*expr*, *dim1*, *dim2*, <*tinc*>, <-*b*>)

expr - expressão

dim1 - ponto inicial (Ex: t=1)

dim2 - ponto final (Ex: t=12)

tinc - incremento

-*b* - contorno exato

- Somatório na área

asum (*expr*, *xdim1*, *xdim2*, *ydim1*, *ydim2*)

xdim1 - dimensão mais a oeste (Ex: lon=0 ou x=1)

xdim2 - dimensão mais a leste (Ex: lon=360 ou x=180)

ydim1 - dimensão mais a sul (Ex: lat=-90 ou y=1)

ydim2 - dimensão mais a norte (Ex: lat=90 ou y=90)

Funções matemáticas

- Outras

`sqrt (expr)`

`pow (expr,p)`

`exp (expr)`

`log10 (expr)`

`log (expr)`

`cos (expr)`

`sin (expr)`

`tan (expr)`

`mag (uexpr,vexpr)`

`hdivg (uexpr,vexpr)`

`hcurl (uexpr,vexpr)`

Funções especiais

- Mudar valores dos dados ausentes

`const` (*expr*, *valor*, -*u*)

- Aplicar uma máscara

`maskout` (*expr*, *mask*)

onde os valores de *mask* forem menores que zero, os valores da *expr* são modificados para valores de dados ausentes. *mask* e *expr* devem, necessariamente, ter o mesmo espaço de grade para `maskout` poder ser utilizado.

Exemplo:

* Fazer a media da temperatura tomando os valores sobre a terra

open temp.ctl

open mask.ctl

→ Máscara de mar-terra, onde os valores sobre mar são negativos

d aave(maskout(p,mask.2(t=1)),lon=0,lon=360,lat=0,lat=90)

Funções especiais

- Interpolação bi-linear entre duas grades

linterp (*variável-fonte, variável-destino*)

Exemplo:

```
open Eta_15kmctl
```

→ Saída do Eta com 15km de resolução

```
open Eta_40kmctl
```

→ Saída do Eta com 40km de resolução

```
define temp15km_interp=linterp(temp.2,temp)
```

→ define a
temperatura do
Eta 40km na
grade de 15km

Funções especiais

Gráfico de linha de uma área

`tloop` (exp)

Ex:

```
'reinit'
```

```
'open Eta_40km.ctl'
```

```
'set lat -22'
```

```
'set lon -45'
```

```
'set t 1 6'
```

```
'd tloop(aave(prec,lon=-45,lon=-40,lat=-25,lat=-20)*1000)'
```

```
'gxprint graf_linha.gif'
```

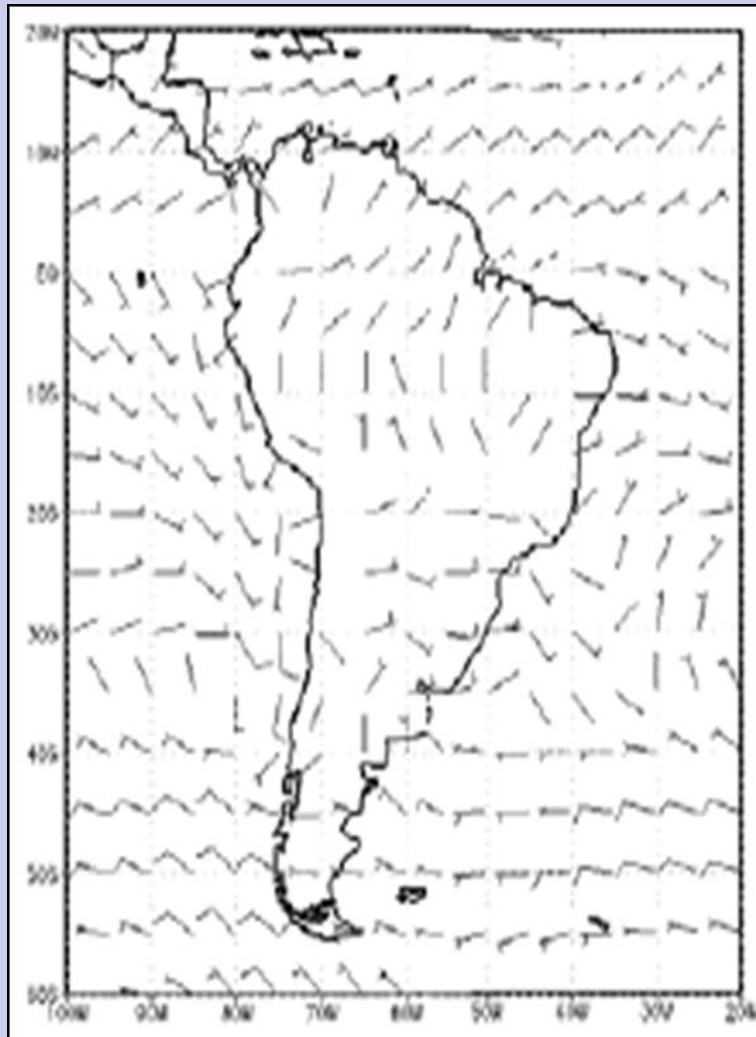
Tipos de saídas gráficas

Comando:

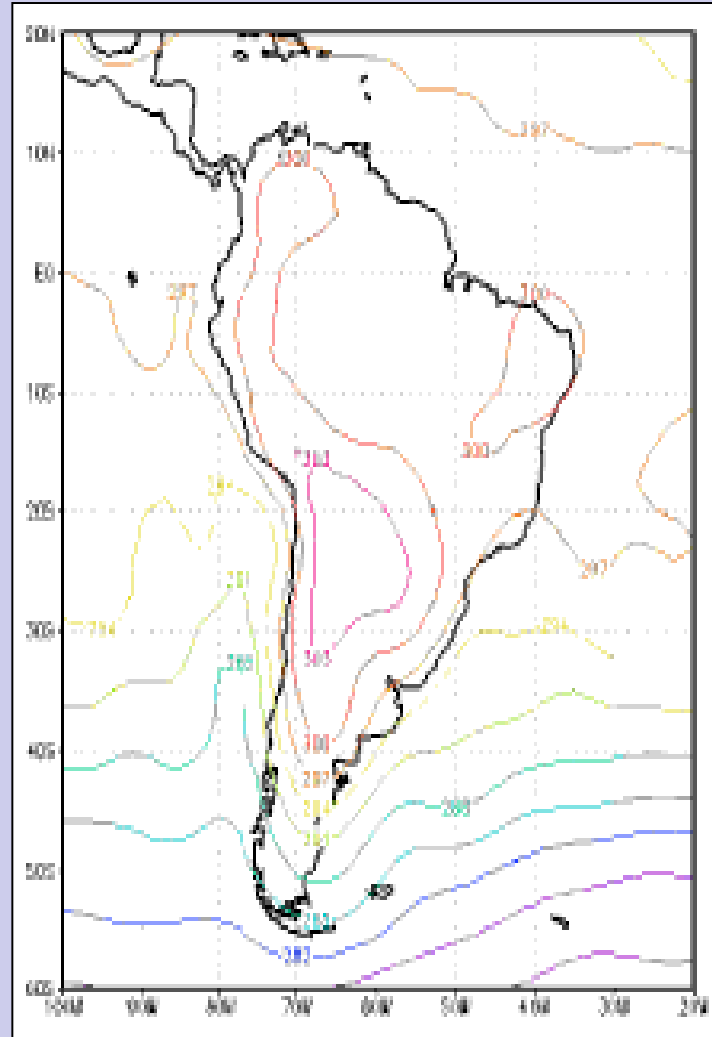
- Sombreado → `set gxout shaded`
- Contorno → `set gxout contour`
- Barbelas de vento → `set gxout barb`
- Flechas do vetor vento → `set gxout vector`
- Linhas de corrente → `set gxout stream`
- Ponto de grade com valor → `set gxout grid`
- Ponto de grade sombreado → `set gxout grfill`
- Barra → `set gxout bar`
- Linha → `set gxout line`

Tipos de saídas gráficas

set gxout barb

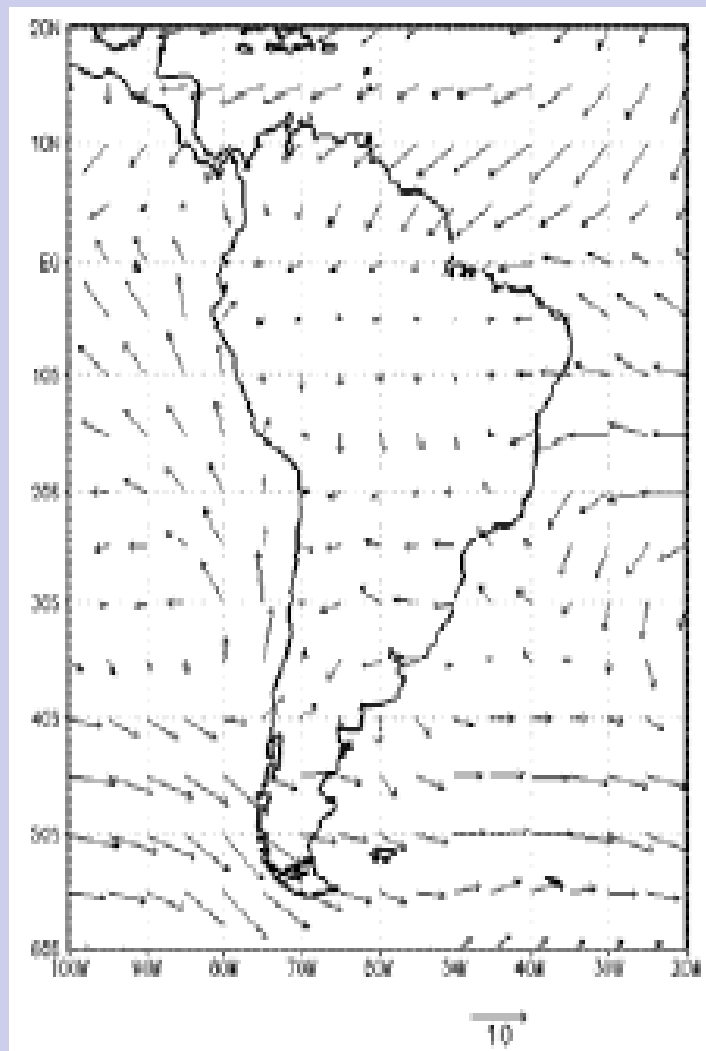


set gxout contour

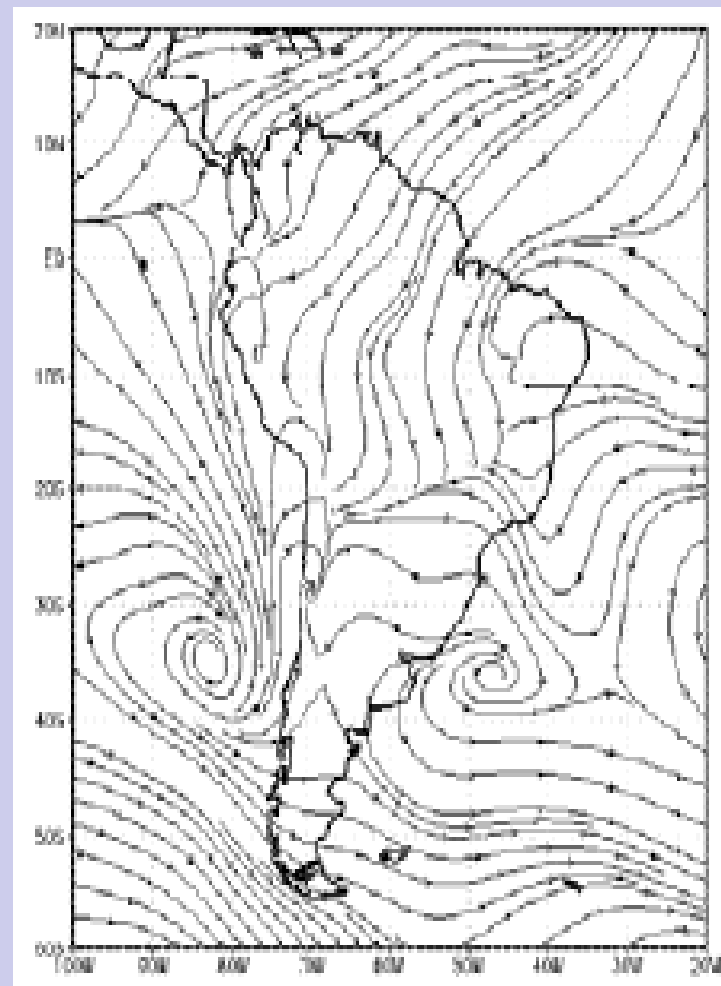


Tipos de saídas gráficas

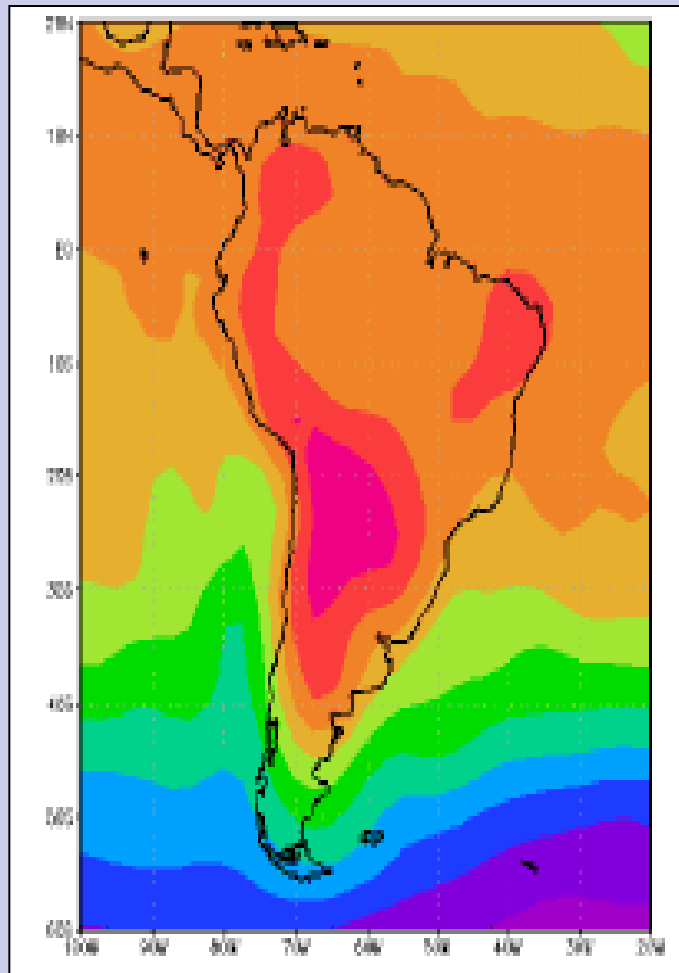
set gxout vector



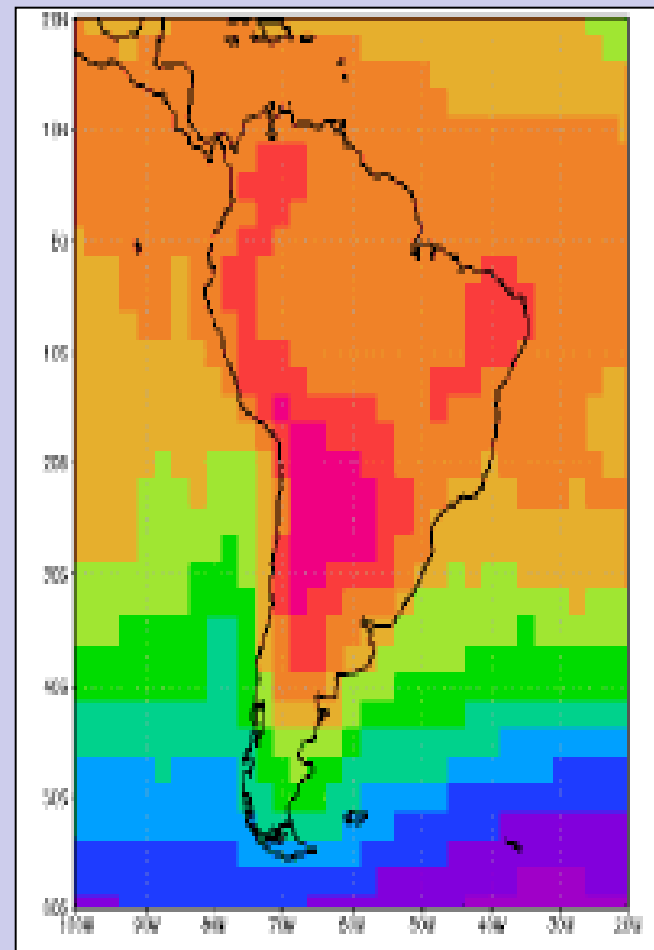
set gxout stream



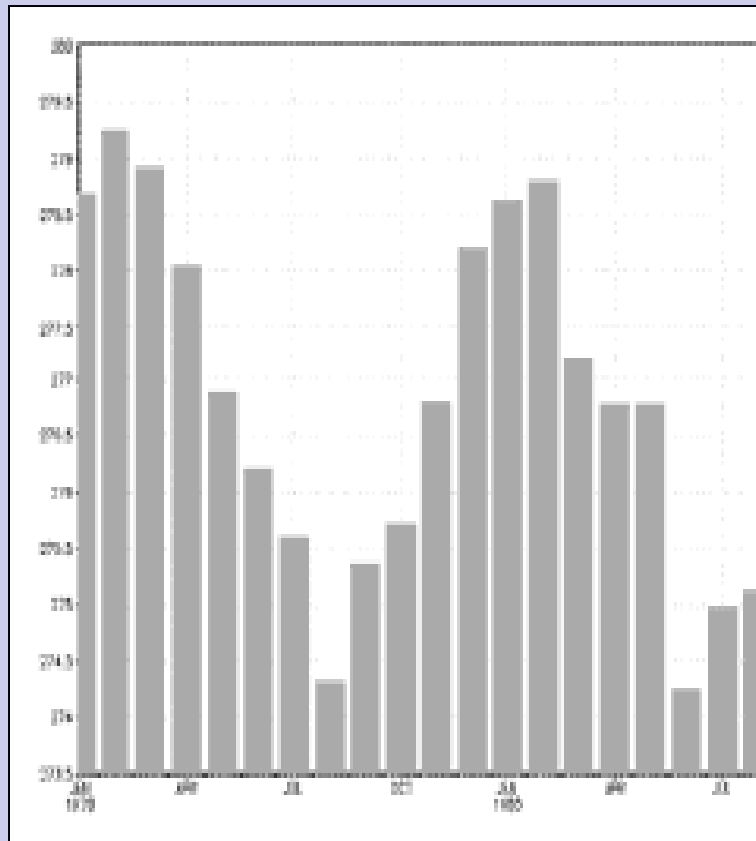
set gxout shaded



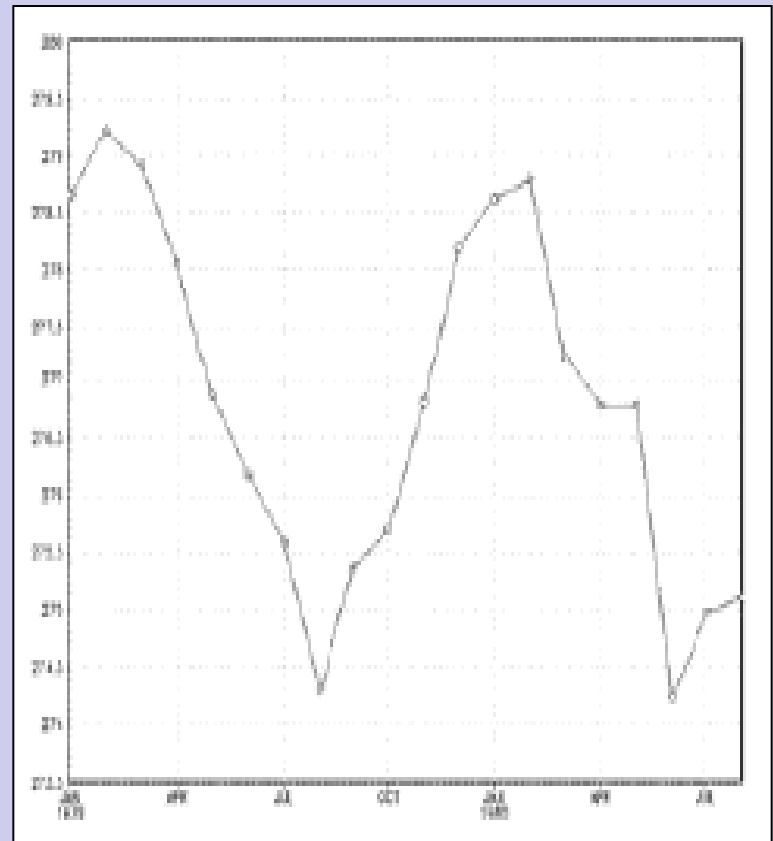
set gxout grfill



set gxout bar



set gxout line



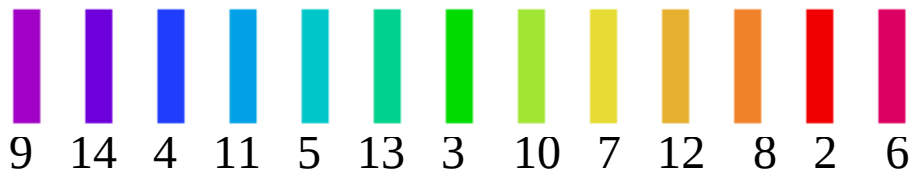
Controlando cores

- Escala de cores

`set ccolor [color]`



`set ccolor rainbow`



- Definir nova cor

`set rgb [número] [R] [G] [B]`

[R] = valor de vermelho (0-255)

[G] = valor de verde (0-255)

[B] = valor de azul (0-255)

[número] = (16-99)

Controlando cores

* These are the BLUE shades

```
set rgb 16 0 0 255
```

```
set rgb 17 55 55 255
```

```
set rgb 18 110 110 255
```

```
set rgb 19 165 165 255
```

```
set rgb 20 220 220 255
```

* These are the RED shades

```
set rgb 21 255 220 220
```

```
set rgb 22 255 165 165
```

```
set rgb 23 255 110 110
```

```
set rgb 24 255 55 55
```

```
set rgb 25 255 0 0
```

```
set clevs lev1 lev2 lev3 ... levN
```

```
set ccols col1 col2 col3 ... colN colN+1
```



Saída de impressão

- Produzir uma figura

`wi` [*nome do arquivo*]

extensões válidas: .GIF .JPG .BMP

`gxprint` [*nome do arquivo*] [*opções*]

GrADS version 2.1

`printim` [*nome do arquivo*] [*opções*]

GrADS

[*opções*] =

{	gif	- imagem gif
	black	- fundo preto
	white	- fundo branco
	xNNN	- tamanho horizontal em NNN pixels
	yNNN	- tamanho vertical em NNN pixels
	-t NN	- faz cor NN transparente

Saída de impressão

- Produzir uma figura

`enable print` [*nome do arquivo.gmf*]

`d` <var ou expr>

`print`

`disable print`

- Programa para converter GMF em GIF

`!gxxgif -r -x` <tamanho em x> `-y` <tamanho em y> `-i` <arq.gmf> `-o` <arq.gif>

Comando query

- Adquirir informação

`query` [*opção*]

ou

`q` [*opção*]

[*opção*] =

ctlinfo	→ mostra informações do arquivo ctl
dims	→ mostra o ambiente dimensionado
file	→ mostra informação do arquivo
files	→ lista os arquivos abertos
pos	→ espera clique do mouse e mostra pos
time	→ mostra o tempo
gxinfo	→ lista atribuições de gráficos

Comando draw

- Escrever título

`draw title [título]`

- Traçar uma linha

`draw line [x1] [y1] [x2] [y2]`

- Desenhar um retângulo

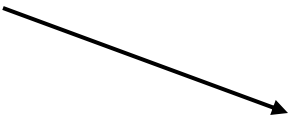
`draw rec [x1] [y1] [x2] [y2]`

- Desenhar um símbolo

`draw mark [marktype] [x] [y] [size]`

- Desenhar uma string

`draw string [x] [y] [string]`



0	none	1	cross
2	open circle	3	closed circle
4	open square	5	closed square
6	X	7	diamond
8	triangle	10	open circle with vertical line
9	none	11	open oval

Controlando os comandos de desenho

- Controlar linha

`set line` [*color*] [*estilo*] [*espesura*]

- Controlar caracteres

`set string` [*color*] [*justificação*] [*espessura*] [*rotação*]

[*justificação*] = tl, tc, tr, t, c, r, bl, bc, br

onde tl (topo esquerda), tc (topo centro)

[*rotação*] = graus

`set strsiz` [*horizontal size*] [*vertical size*]

`set font` [*número*]

[*número*] = 1 a 5

Grads scripts

- Criar um script (extensão **.gs**)

nedit [nome do script].gs

gedit [nome do script].gs

vi [nome do script].gs

- Comentários: ***** (asterisco no início)
- Comandos Grads: sempre entre ‘ ’ (aspas simples)
- Executar o script

ga-> **run** [script].gs

ou de fora do grads: **gradsc -pc “run [script]. gs”**

Grads scripts

- `say / prompt` → Apresentar informação ou fazer questão

Exemplo:

```
frase = "Peter Pan, o voador"
```

```
say frase
```

```
say `Ela disse ele é `frase
```

Resultado:

Peter Pan, o voador

Ela disse ele é Peter Pan, o voador

- `pull` → Fornecer informação para o script

Exemplo:

```
prompt 'Entre min e max latitudes: '
```

```
pull minlat maxlat
```

```
'set lat 'minlat%' '%maxlat
```

Grads scripts (Controle de fluxo)

- if / else / endif → Controlar a execução

Exemplo1:

```
if (i = 10)
```

```
  j = 20
```

```
else
```

```
  j = 30
```

```
endif
```

Exemplo2:

```
if (i = 10) ; j = 20 ; endif
```

Grads scripts (Controle de fluxo)

- while / endwhile / break → Controlar a execução

Exemplo:

```
count = 1
```

```
while (count < 10)
```

```
'set t 'count
```

```
say count
```

```
if (count = 6) ; break ; endif
```

```
count = count + 1
```

```
endwhile
```

Grads Scripts

- Operadores

	lógico OU
&	lógico E
=	igual
!=	não igual
>	maior que
>=	maior ou igual que
<	menor que
<=	menor ou igual que
%	concatenação
+	adição
-	subtração
*	multiplicação
/	divisão

Grads Scripts

- Funções Intrínsecas

`sublin` (*result*,*n*)

O resultado é a *n-ésima* linha de um conjunto de caracteres *result*.

`subwrd` (*result*,*n*)

O resultado é a *n-ésima* palavra do conjunto de caracteres *result*.

`substr` (*result*,*i*,*c*)

O resultado é o sub-conjunto de caracteres do conjunto de caracteres inicia na localização *i* e tem o comprimento *c*.

Grads Scripts

- Scripts prontos

<http://cola.gmu.edu/grads/gadoc/gadocindex.html> - **GrADS Script Library**

basemap.gs	Overlays a land or ocean mask that exactly fits the coastal outlines. Requires the following supplemental data files: lpoly_lowres.asc and lpoly_mres.asc and lpoly_hires.asc opoly_lowres.asc and opoly_mres.asc and opoly_hires.asc See instructions in script header for using lpoly_US.asc to mask out non-US areas.
box_and_whisker.gs	Demonstrates how to use gxout bar and errbar to draw a box and whisker plot
cbar.gs and cbarn.gs cbarm.gs	Scripts to draw a long rectangular color legend next to shaded plots. cbar.gs is the original version -- just the filled rectangles with labels cbarn has some added features and arguments -- it draws outlines and triangular endpoints cbarm will look better if using 30+ colors -- labels are drawn at appropriate intervals
cbarc.gs	Draws a small fan-shaped color legend in the corner of shaded plots.
cbar_l.gs cbar_line.gs cbar_line2.gs	Scripts to draw a legend for line graphs.

Escrevendo arquivos de saída

- Escrevendo um arquivo binário

set gxou fwrite

set fwrite [-be ou -le] [-sq ou -st] [-ap ou -cl] [*fname*]

fname output filename (default = grads.fwrite)

-be output data byte ordering is big endian

-le output data byte ordering is little endian

-sq output data format is sequential

-st output data format is stream (default)

-ap output data is appended to existing file

-cl output data replaces existing file if it exists (default)

d [*expr*]

disable fwrite

Escrevendo arquivos de saída

Exemplo: Escrevendo um arquivo binário

➤ nedit script1.gs

```
'open Eta_40km.ctl'
```

```
'set gxout fwrite'
```

```
'set fwrite Eta_40km_prec.dat'
```

```
' set x 1'
```

```
' set y 1'
```

```
tempo=1
```

```
while(tempo<=6)
```

```
' set t ' tempo
```

```
'd aave(prec,lon=-45,lon=-40,lat=-25,lat=-20)'
```

```
tempo=tempo+1
```

```
endwhile
```

```
'disable fwrite'
```

```
'quit'
```

Exemplo: Criar um ctl

➤ nedit Eta_40km_prec.ctl

```
dset Eta_40km_prec.dat
```

```
undef -9999.
```

```
xdef 1 linear 1 1
```

```
ydef 1 linear 1 1
```

```
zdef 1 levels 1000
```

```
tdef 6 Linear 00z19mar2019 1hr
```

```
vars 1
```

```
prec 0 99 Precipitação
```

```
endvars
```

Escrevendo arquivos de saída

- Escrevendo um arquivo no formato ASCII (somente em scripts)

`d [expr]`

`valor= sublin(result,4)`

`write [filename, valor]`

ou

`write [filename, valor, append]` → para acrescentar em um
arquivo existente

Respostas do comando:

0 - ok

1 - open error

8 - file open for read

Escrevendo arquivos de saída

Exemplo: Escrevendo um arquivo ascii (.txt)

```
➤ nedit script2.gs  
'open Eta_15km.ctl'  
'set lat -25'  
'set lon -45'  
'd tp2m'  
linha=sublin(result,1)  
valor=subwrd(linha,4)  
say linha  
say valor  
write(valor.txt, valor)
```