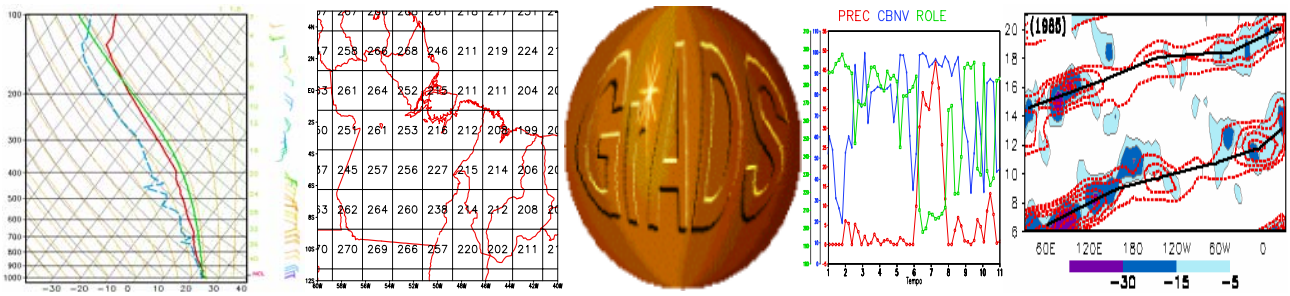
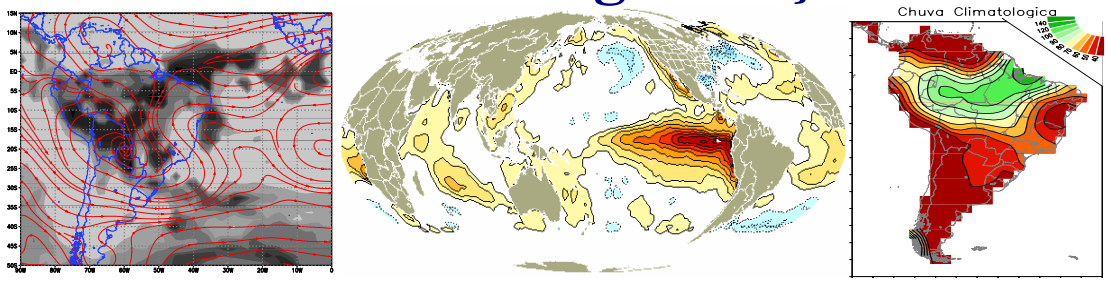
UNIVERSIDADE FEDERAL DO PARÁ

CENTRO DE GEOCIÊNCIAS

DEPARTAMENTO DE METEOROLOGIA



GrADS – Grid Analysis and Display System
Fundamentos e Programação Básica



POR

EVERALDO BARREIROS DE SOUZA

Pesquisador Dr. do CNPq / DM-CG-UFPa

everaldo@ufpa.br

APOSTILA DO CURSO DE EXTENSÃO
ORGANIZADO PELA COORDENAÇÃO DO COLEGIADO DE METEOROLOGIA

BELÉM – PA
9 A 13 DE FEVEREIRO DE 2004

ÍNDICE

0. NOTA DO AUTOR

Primeiramente, o autor expressa seus agradecimentos a coordenação do colegiado de meteorologia, especificamente ao Prof. João Batista pelo convite e ao Prof. José de Paulo pelo apoio e organização deste curso. A presente apostila foi preparada para ministrar o curso **"GrADS – Fundamentos e Programação Básica"** aos professores e alunos concluintes do curso bacharelado em meteorologia. Todos os materiais e arquivos digitais usados neste curso encontram-se no CD-ROM em anexo. Utiliza-se neste curso o **Win32e GrADS**, o qual deverá ser **instalado** em seu PC (ver detalhes na secção 2.1) juntamente com o diretório **grads-everaldo** contendo os arquivos digitais dos dados usados nos exercícios propostos explicados em sala de aula.

Divirta-se usando o GrADS em suas tarefas de operação e pesquisa meteorológica!!!

Everaldo B. de Souza <*everaldo@ufpa.br*>
Instrutor do curso "GrADS – Fundamentos e Programação Básica"
Belém, sexta-feira, 13 de fevereiro de 2004.

1. O SOFTWARE

1.1. O que é o GrADS ?

O GrADS – Sistema de Visualização e Análise de Dados em Pontos de Grade – é um software interativo utilizado nas tarefas de acesso, manipulação e visualização de dados geofísicos em geral. O GrADS trabalha com matrizes de dados nos formatos BINÁRIO, GRIB, NetCDF ou HDF-SDS, nas quais as variáveis podem possuir até 4 dimensões (longitude, latitude, níveis verticais e tempo) especificadas por um arquivo descritor.ctl. Atualmente, o GrADS é o software mais utilizado nos centros operacionais e de pesquisa meteorológica espalhados pelo mundo, inclusive no Brasil. Este software foi originalmente desenvolvido pelo pesquisador Brian Doty (doty@cola.iges.org) no COLA (grads.iges.org/cola.html) dentro da Universidade de Maryland no final da década de 80. Sua distribuição é totalmente livre e gratuita através de sua página oficial: http://grads.iges.org/grads/index.html. As matrizes de dados podem conter uma ou mais variáveis dispostas numa grade regular, ou não linear, ou gaussiana, ou em pontos de estações ou de resolução variável. As variáveis podem ser plotadas e combinadas usando vários tipos de gráficos, os quais podem ser gravados em formato PostScript ou diversos formatos de imagem gráfica (PNG, GIF, JPEG, etc). O GrADS possui uma interface programável (scripting language) com a qual é possível se desenvolver sofisticadas análises, cálculos de variáveis derivadas e aplicações de visualização automática (interfaces gráficas com buttons e dropmenus clicáveis). Dentro dos scripts é possível se desenvolver a interatividade com funções, expressões ou rotinas externas escritas com outras linguagens de programação (FORTRAN, C, C++, UNIX Shell, etc) e também com linhas de comando do sistema operacional (MS-DOS, Windows, LINUX, UNIX). As versões atuais trazem uma grande variedade de funções intrínsecas (funções do próprio GrADS), mas o usuário também pode adicionar sua própria função usando rotinas externas desenvolvidas em FORTRAN ou outra linguagem. O GrADS pode ser executado em modo batch e, portanto os scripts podem ser usados para realizar tarefas automáticas sem a necessidade da presença direta do usuário.

1.2. Downloading o GrADS

Na página de download oficial do GrADS (http://grads.iges.org/grads/downloads.html) encontram-se disponíveis os arquivos executáveis pré-compilados (arquivos binary), o código fonte

e os conjuntos de dados e utilitários suplementares (arquivos de mapas, fonte, etc) necessários para a instalação e execução do GrADS. A tabela abaixo contém os links para fazer o download via FTP da versão mais recente do GrADS disponível para diversas plataformas e sistemas operacionais. Cada versão vem acompanhada de um readme contendo as instruções de instalação.

Hardware /Operating System	GrADS 1.8s11 (Full Distribution: source, binaries, data)	GrADS 1.8s11 (Just the binary executables)
DEC	**alpha** (full)	**alpha** (binaries)
Intel / LINUX	**linux** (full)	**linux** (binaries)
SUN	**sol55** (full)	**sol55** (binaries)
Macintosh OSX	**darwin** (full)	**darwin** (binaries)
SGI	**irix6** (full)	**irix6** (binaries)
IBM / AIX	**aix** (full)	**aix** (binaries)
MS Windows	**xwin32** (Requires an X server; Getting xwin32 started) **win32e** (Uses native windows; Getting win32e started)	

1.3. Documentação

A documentação online e todos os manuais encontram-se disponíveis na página http://grads.iges.org/grads/gadoc

1.4. Suporte e Lista de Discussões

Existe uma lista de usuários do GrADS efetivamente ativa, na qual é possível compartilhar informações, saber dos refinamentos e desenvolvimentos recentes, versões novas, bem como principalmente ajudar na solução de problemas e dúvidas dos usuários do GrADS em geral.

Para fazer parte da lista do GrADS, envie um e-mail para o endereço

gradsusr-request@list.cineca.it

e forneça seu endereço, instituição, etc.

Para ver o arquivo online da lista do GrADS acesse o endereço

http://dao.gsfc.nasa.gov/grads_listserv/

2. FUNDAMENTOS E COMANDOS BÁSICOS

2.1. Instalando o Win32e GrADS em seu PC Windows

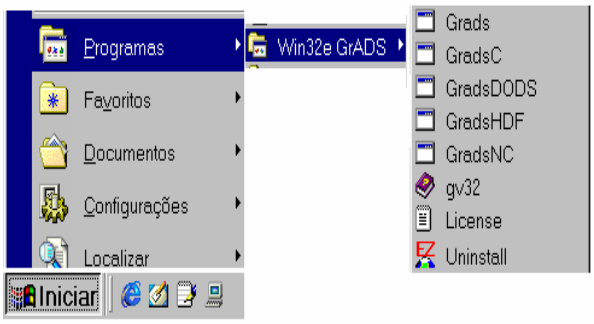
Neste curso, utilizaremos o Win32e GrADS (uma versão do GrADS que trabalha especificamente na plataforma PC 32-bit Windows sem a necessidade de instalar um X-server, a qual foi desenvolvida pelo Dr. Arlindo da Silva <dasilva@dao.gsfc.nasa.gov>).

No CD-ROM que acompanha esta apostila encontra-se disponível o arquivo

que faz a instalação do Win32e GrADS (versão 1.8sl11) em seu PC, seguindo os seguintes procedimentos:

- Coloque o CD-ROM no seu drive de CD;

- Clicar em Meu Computador, depois no seu drive de CD ;
- Localizar e depois clicar duas vezes no arquivo de instalação do GrADS;
- Siga os passos/informações de instalação;
- Após a instalação, o programa insere no menu iniciar/programas a pasta com os arquivos de execução do GrADS, conforme a ilustração ao lado;
- É possível também abrir o GrADS a partir do prompt do MS-DOS (altere seu autoexec.bat colocando no path o diretório onde encontram-se os arquivos executáveis do GrADS;



- Copiar também o diretório **grads-everaldo** para dentro do seu disco C:/, o qual contém os arquivos de dados necessários para fazer os exemplos e exercícios práticos a serem explicados em sala de aula;
- Pronto! Agora você está apto a fazer este curso e aprender os conceitos básicos do GrADS.

2.2. Os Arquivos de Dados.dat e Descritor.ctl

Basicamente, o GrADS trabalha com dois arquivos principais:

- **o arquivo de dados (por exemplo, dados.dat)**
- **e o arquivo descritor (por exemplo, descritor.ctl)**

O dados.dat deve estar nos formatos BINÁRIO, GRIB, NetCDF ou HDF-SDS. O descritor.ctl é um arquivo tipo texto, no qual descrevem-se todas as especificações da dimensão dos dados.dat. Um exemplo simples de arquivo descritor encontra-se abaixo:

```
DSET vento.dat
TITLE Dados de Vento em Ar Superior
UNDEF -99999
XDEF 80 LINEAR -140.0 1.0
YDEF 50 LINEAR   20.0 1.0
ZDEF 5 LEVELS 1000 850 500 300 100
TDEF 4 LINEAR 0Z10apr1991 12hr
VARS 2
u 5 0 componente u do vento
v 5 0 componente v do vento
ENDVARS
```

Significado de cada linha do arquivo descritor:

`DSET vento.dat`	Especifica o nome do arquivo de dados
`TITLE Dados de Vento em Ar Superior`	Título do conjunto de dados
`UNDEF -99999`	Valores indefinidos (ignorados na plotagem)
`XDEF 80 LINEAR -140.0 1.0`	Especifica a dimensão X (longitude)

`XDEF 80 LINEAR -140.0 1.0`
- `1.0` → espaçamento em pontos de grade
- `-140.0` → ponto (longitude) inicial
- `LINEAR` → varia linearmente
- `80` → número de pontos na direção x

```
YDEF 50 LINEAR   20.0 1.0
```

Especifica a dimensão Y (latitude)

- 1.0 → espaçamento em pontos de grade
- 20.0 → ponto (latitude) inicial
- LINEAR → varia linearmente
- 50 → número de pontos na direção y

```
ZDEF 5 LEVELS 1000 850 500 300 100
```

Especifica a dimensão Z (níveis verticais)

- 1000 850 500 300 100 → os cinco níveis verticais
- 5 → número de níveis verticais

```
TDEF 4 LINEAR 0Z10apr1991 12hr
```

Especifica a dimensão T (tempo)

- 12hr → espaçamento temporal de 12 em 12 horas
- 0Z10apr1991 → tempo inicial
- LINEAR → varia linearmente
- 4 → número de tempos

```
VARS 2
u 5 0 componente u do vento
v 5 0 componente v do vento
```

Especifica o número de variáveis contidas no arquivo vento.dat

- componente u do vento / componente v do vento → texto com descrição de cada variável
- 0 → código de unidades (dependo do formato dos dados)
- 5 → número de níveis verticais para cada variável
- u / v → abreviação para cada variável

```
ENDVARS
```

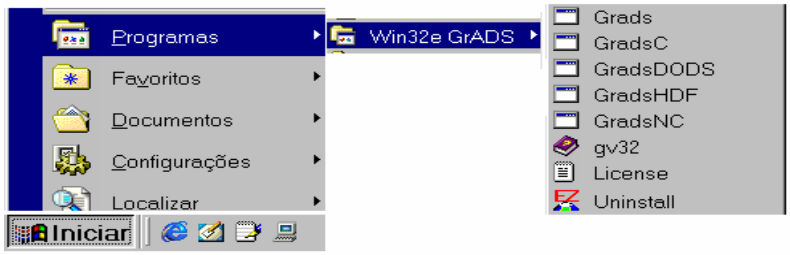
Fim do descritor.ctl e fim das especificações das variáveis

Observação: A descrição completa dos componentes do arquivo descritor para os diversos formatos de dados encontra-se no Apêndice A1.

2.3. Executando o GrADS (Tela Inicial)

O GrADS pode ser iniciado diretamente com o seu mouse clicando no menu iniciar/ programas/ Win32e GrADS/ Grads
(ver ilustração ao lado)

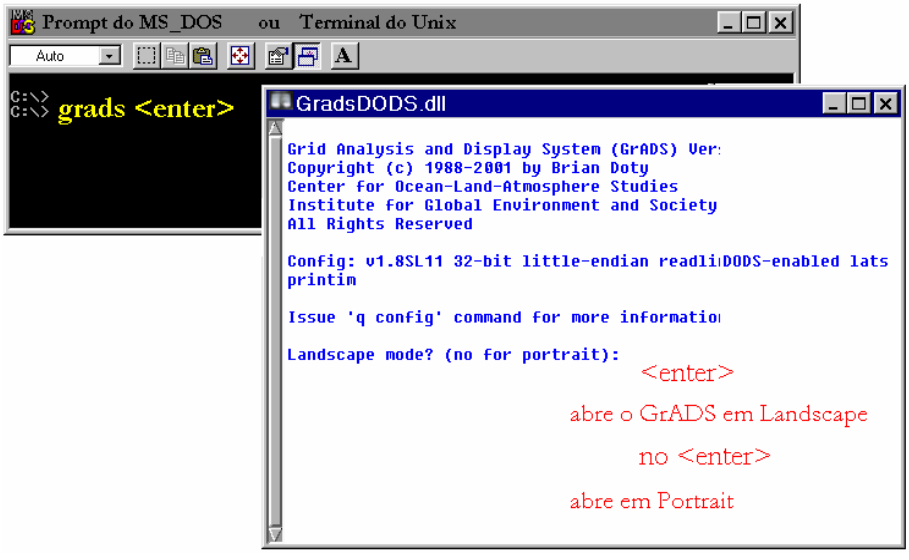
Abrindo o GrADS a partir do menu iniciar/programas



Ou a partir de uma janela (prompt do MS-DOS ou Terminal do Unix/Linux) digitando-se o comando:

grads <enter>
(ver ilustração ao lado)

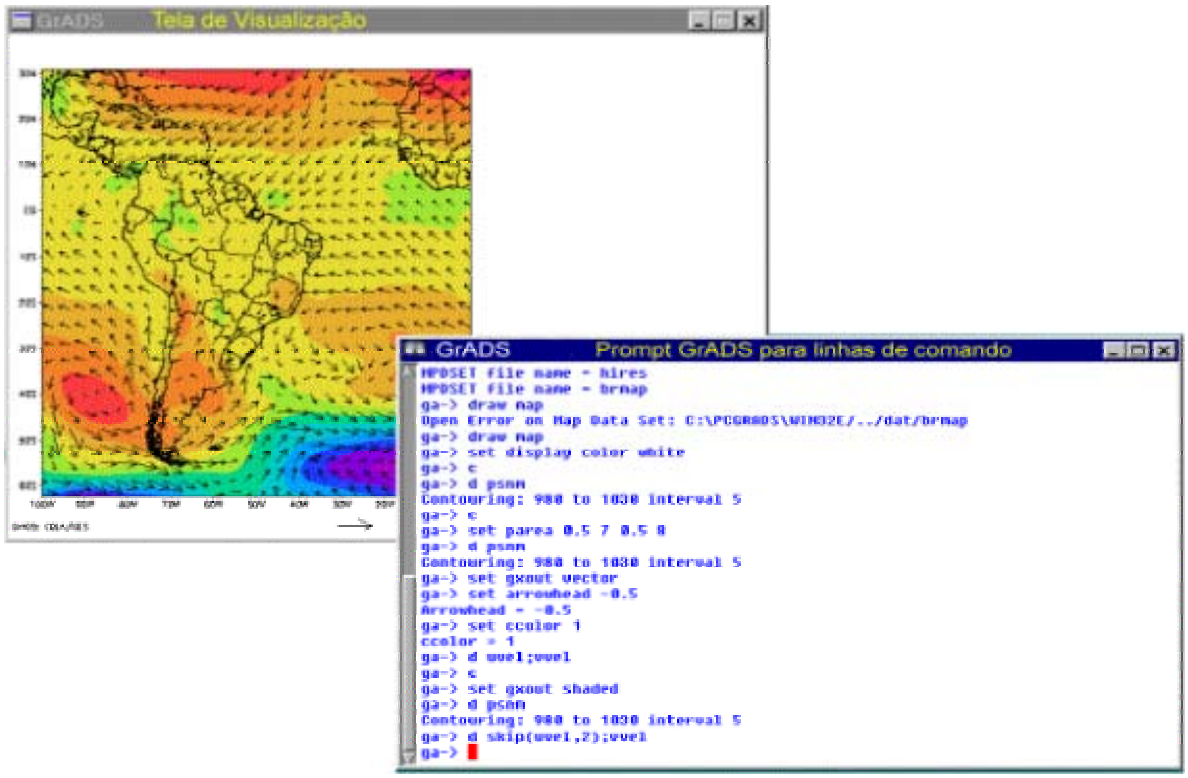
Abrindo o GrADS a partir do prompt do MS_DOS ou Terminal do Unix/Linux



Após o comando, aparece o texto de versão, copyright, etc e escolhe-se a opção de janela de visualização no tamanho **Landscape**
(opção default, basta dar o <enter>)
ou **Portrait**
(digite no <enter>)
(ver ilustração ao lado)

Em seguida, duas janelas são abertas, conforme ilustração mostrada abaixo:

- uma **tela de visualização** na qual são plotados os gráficos e mapas;
- e outra tela que é o **prompt do GrADS** na qual são digitadas as linhas de comandos, conforme ilustra a figura abaixo.



Dica: A tela de visualização do GrADS abre sempre com o fundo preto, o que, por vezes, dificulta a interpretação de certos gráficos. Para mudar o fundo da tela de visualização para a cor branca, faça o seguinte:

ga> set display color white
ga> clear

Observação: Outros comandos de abertura encontram-se listados abaixo

grads –l	abre o GrADS em modo landscape
grads –p	abre o GrADS em modo portrait
grads –b	executa o GrADS em modo batch (nenhuma janela é aberta)
grads -c "linha de comando do GrADS"	abre o GrADS e executa a linha de comando entre aspas

Estas opções podem ser usadas em combinações, do tipo:

grads –lc "open exemplo.ctl" ou **grads –bpc "run scripts.gs"**

2.4. Abrindo e Visualizando os Dados

Dentro do prompt do GrADS, o comando de abertura do arquivo descritor (que por sua vez controla o arquivo de dados) é feito da seguinte forma:

ga> open exemplo.ctl

```
Scanning description file: exemplo.ctl
Data file exemplo.grb is open as file 1
LON set to –150 0
LAT set to –62.486 30.77
LEV set to 1000 1000
Time values set 2004:2:5:6 2004:2:5:6
```

} informações que aparecem quando da abertura do CTL...

ga>

O comando para visualizar uma variável é feito da seguinte forma:

ga> display nomedavariável

ou simplesmente:

ga> d nomedavariável

Para sair do GrADS, basta digitar o comando:

ga> quit

2.5. O Comando set

O comando **set** especifica "quando", "onde" e "como" as variáveis serão plotadas. Por exemplo:

ga> set t 1 **ga> set lat –20 -10** **ga> set gxout line**

2.6. Manipulando as Dimensões

A manipulação das dimensões é feita usando o comando **set**, conforme exemplos abaixo:

ga> set lat valordaLAT1 valordaLAT2	Especifica a grade entre as latitudes valordaLAT1 e valordaLAT2; se valordaLAT2 não for especificado, tem-se a latitude fixada no ponto da valordaLAT1
ga> set y valordeZ1 valordeZ2	Idem acima
ga> set lon valordaLON1 valordaLON2	Especifica a grade entre as longitudes valordaLON1 e valordaLON2; se valordaLON2 não for especificado, tem-se a latitude fixada no ponto da valordaLON1
ga> set x valordeZ1 valordeZ2	Idem acima
ga> set lev valordeZ1 valordeZ2	Especifica a grade entre os níveis verticais valordeZ1 e valordeZ2; se valordeZ2 não for especificado, tem-se o nível vertical fixo em valordeZ1
ga> set z valordeZ1 valordeZ2	Idem acima
ga> set t valordeT1 valordeT2	Especifica a grade entre os tempos valordeT1 e

valordeT2; se valordeT2 não for especificado, tem-se o tempo fixo em valordeT1

ga> set time valordeT1 valordeT2 Idem acima, porém a sintaxe de valordeT1 e valordeT2 deve ser na forma: 00z09feb2004

Observações:

- Os valores da LAT do Hemisfério Sul e LON do Hemisfério Oeste são precedidos do sinal negativo.
- O GrADS considera a dimensão Y variando de sul para norte e a dimensão X variando de oeste para leste. Portanto, quando da especificação das mesmas, é necessário fazer o set primeiro da LAT (LON) mais ao sul (oeste).
  Por exemplo:
  ga> set lat -30 -5
  ga> set lon -80 -20

2.7. Outros Comando Básicos

O comando **query** ou **q** serve para obter informações sobre os arquivos de dados (nome das variáveis, etc), sobre dimensões, sobre posições de tela e geográfica, sobre estatísticas em geral, etc. Por exemplo:

ga> q file Especifica as informações gerais do arquivo descritor

```
File 1: Dados de Vento em Ar Superior
Descriptor: vento.ctl
Type: gridded
Xsize = 80  Ysize = 50  Zsize = 5  Tsize = 4
Number of variables = 2
u 5 0 componente u do vento [m / seg]
v 5 0 componente v do vento [m / seg]
```

} resultados do commando q file

Observação: Se houver vários arquivos descritores abertos, usa-se:
ga> q files ou **ga> q file n** para o **n** CTL aberto

ga> q dims Especifica as dimensões correntes

```
Default file number is: 1
X is varying    Lon = -150 to 0          X = 1 to 81
Y is varying    Lat = -62 to 30          Y = 1 to 51
Z is fixed      Lev = 1000               Z = 1
T is fixed      Time = 06z05FEB2004      T = 1
```

resultados do commando q dims

ga> clear Limpa a tela de visualização

ga> c Idem acima

ga> reinit Reinicia o GrADS; fecha todos ctl abertos

ga> reset Reinicia o GrADS; porém sem fechar os ctl

ga> ! linha-de-comando Executa linha de comando do sistema operacional

ga> help help básico

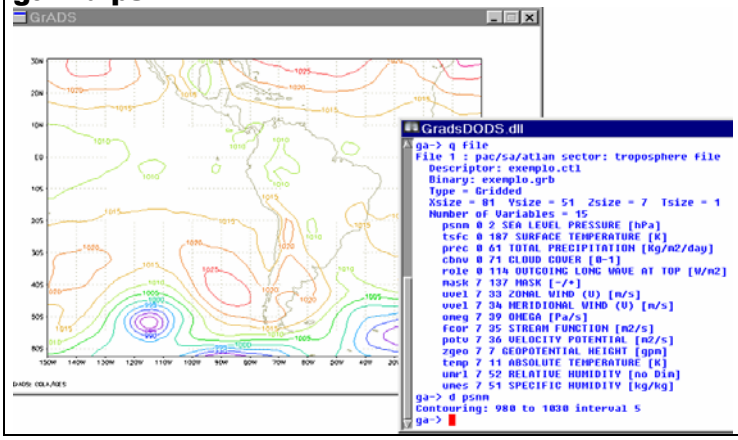
2.8. Exemplos e Exercícios Básicos

Os exemplos e exercícios básicos serão conduzidos com base no `exemplo.ctl` e seu respectivo arquivo de dados, os quais encontram-se no diretório c:/grads-everaldo.

Exemplo 1:
Abra o GrADS em modo Portrait e plote a variável pressão ao nível médio do mar

Dentro do prompt do GrADS, digite:

ga> set display color white
ga> c
ga> open \grads-everaldo\exemplo.ctl
ga> q file
ga> d psnm

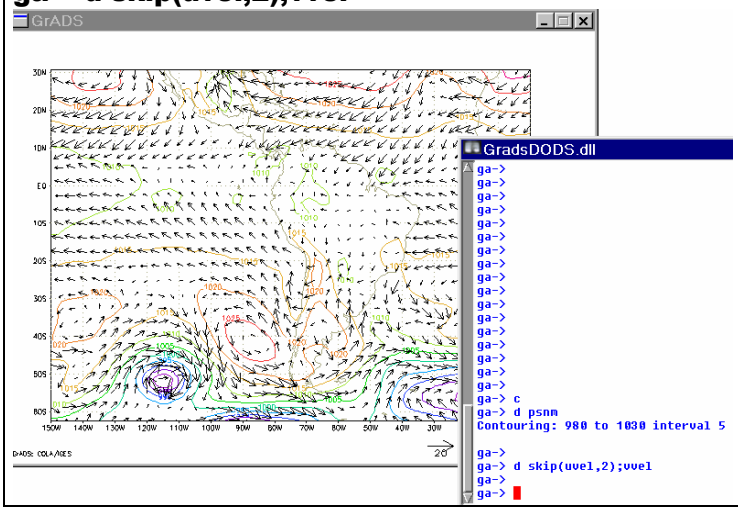


Exercício proposto 1:

Abra o GrADS em Landscape e plote o campo de precipitação

Exemplo 2:
Plotando duas variáveis sobrepostas (pressão e vento horizontal)

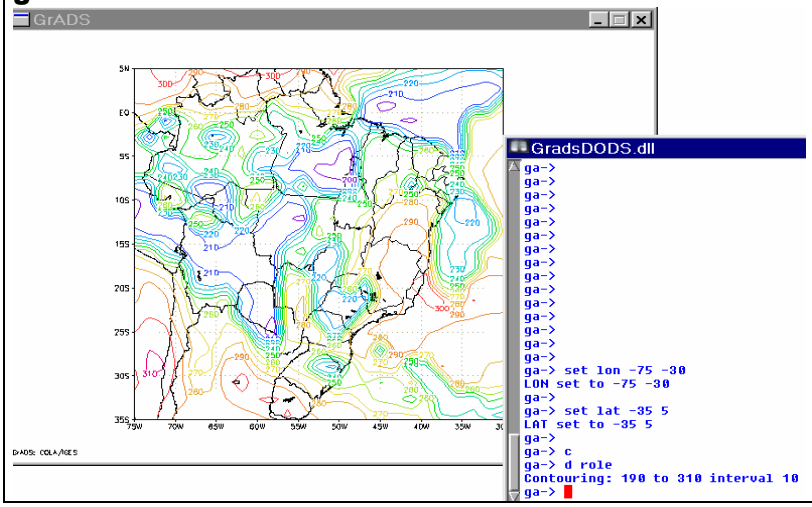
ga> c
ga> d psnm
ga> d uvel;vvel
ou
ga> d skip(uvel,2);vvel



Exercício proposto 2:
Plote o campo de cobertura de nuvens sobreposto ao campo de vento horizontal

Exemplo 3:
mapa de radiação de onda longa redimensionado para a região do Brasil

ga> c
ga> set mpdset hires brmap
ga> q dims
ga> set lat –35 5
ga> set lon –75 -30
ga> d role



Exercício proposto 3:
Plote o mapa de umidade relativa sobre o estado do Pará

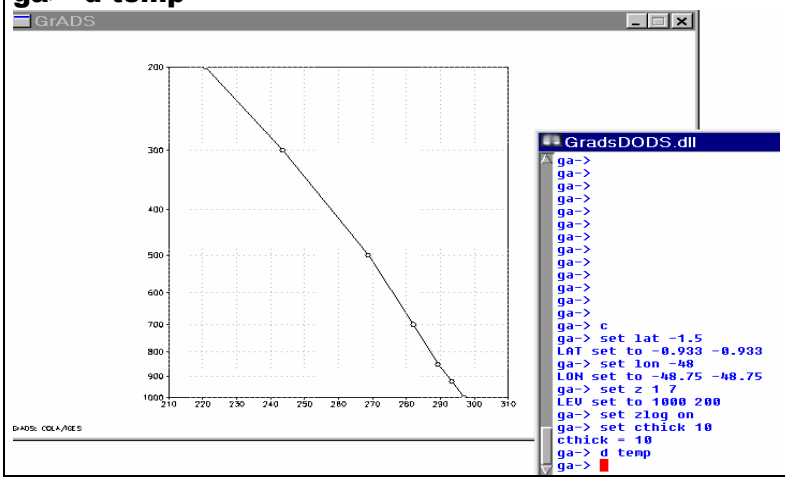
Exemplo 4:
Mapa do geopotencial em 500 hPa

ga> c
ga> set lev 500
ga> d zgeo

Exercício proposto 4:
Plote o vento horizontal em 200 hPa

Exemplo 5:
perfil vertical de temperatura sobre o ponto centrado em Belém

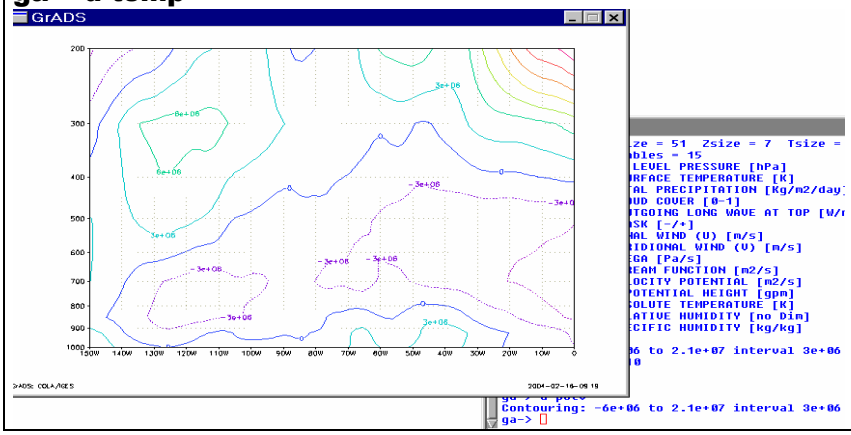
ga> c
ga> set lat –1.5
ga> set lon –48
ga> set z 1 7
ga> set zlog on
ga> d temp



Exercício proposto 5:
Plote o perfil vertical de umidade específica sobre o ponto centrado em Manaus

Exemplo 6:
Perfil vertical zonal de velocidade potencial ao longo da faixa equatorial (secção longitude x altitude)

ga> reset
ga> set lat 0
ga> set z 1 7
ga> set zlog on
ga> d temp



Exercício proposto 6:
Plote a secção vertical meridional (latitude x altitude) de altura geopotencial ao longo da longitude de Belém

Os dois exemplos a seguir são realizados com base nos arquivos gpcp-1983.ctl (precipitação pentadal do GPCP para o ano de 1983).

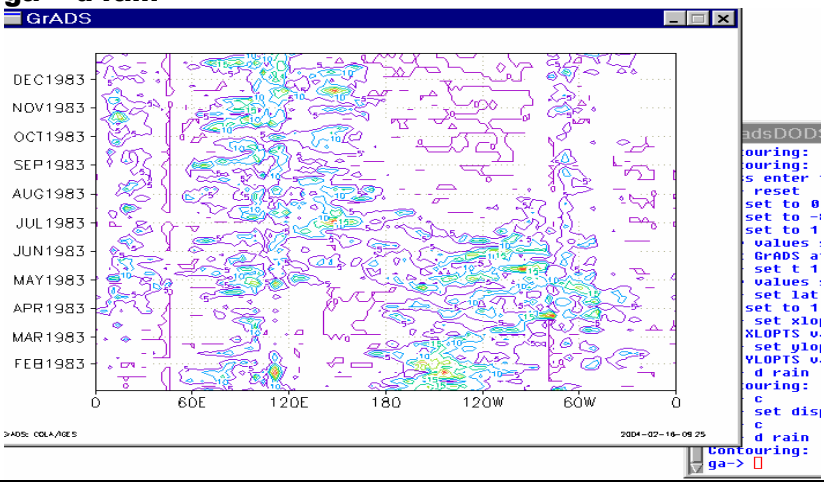
Exemplo 7:
Animação temporal de chuva observada no Brasil durante janeiro a junho de 1983

ga> reinit
ga> open \grads-everaldo\gpcp-1983.ctl
ga> set lat –35 10
ga> set lon –75 -30
ga> set time jan1983 jun1983
ga> d rain

Exercício proposto 7:
Faça a animação da chuva observada sobre a região da Amazônia entre os meses de julho a dezembro de 1983

Exemplo 8:
Diagrama de hovmöller da chuva observada durante o ano de 1983 ao longo do globo e sobre a linha do equador

ga> set t 1 last
ga> set lat 0
ga> d rain



Exercício proposto 8:
Faça o diagrama de hovmöller da chuva observada em 1998 ao longo das longitudes do Brasil especificamente sobre a latitude de Belém.

3. PLOTANDO GRÁFICOS

3.1. Tipos de Gráficos

Existem diversas opções de gráficos. Por default, se o usuário não especificar nenhum tipo de gráfico, tem-se a plotagem do tipo line (para dados com 2 dimensões) e do tipo contour (para gráficos com 3 dimensões).

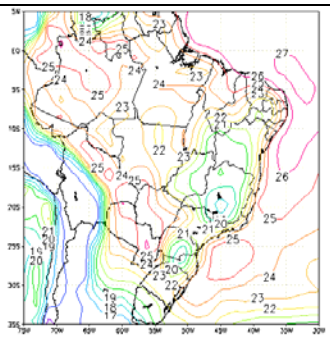
A linha de comando para escolher o tipo de gráfico é:

ga> set gxout tipo_de_grafico

No GrADS tem-se várias opções (tipos) de gráficos, conforme exemplos a seguir:

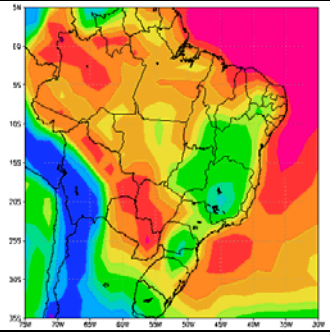
Exemplo 9: Contornos (isolinhas)

```
ga> open \grads-everaldo\exemplo.ctl
ga> set display color white
ga> c
ga> set mpdset hires brmap
ga> set lat –35 5
ga> set lon –75 -30
ga> set gxout contour
ga> d temp-273
```



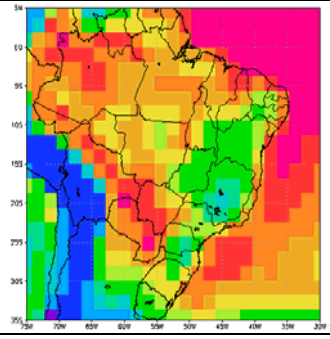
Exemplo 10: Contornos sombreados (faixas de cores)

```
ga> c
ga> set gxout shaded
ga> d temp-273
```



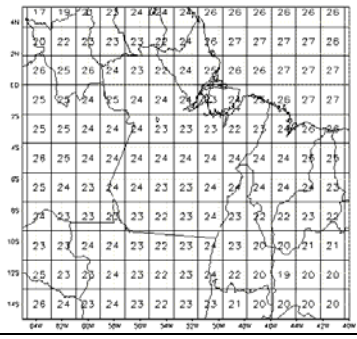
Exemplo 11: Idem, mas com sombra nos pontos de grade

```
ga> c
ga> set gxout grfill
ga> d temp-273
```



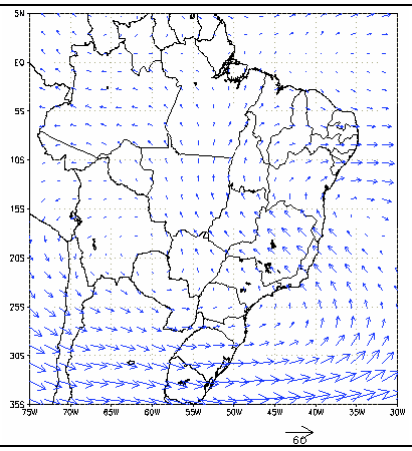
Exemplo 12: Valores nos pontos de grade

```
ga> c
ga> set gxout grid
ga> d temp-273
```



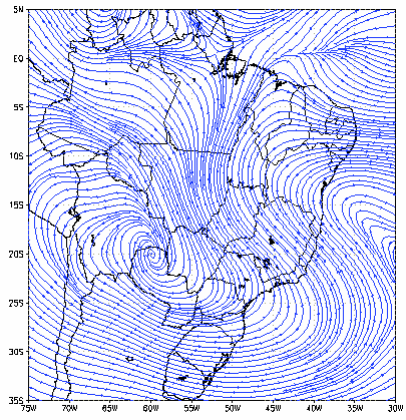
Exemplo 13: Vetores (setas)

```
ga> c
ga> set gxout vector
ga> d uvel;vvel
```



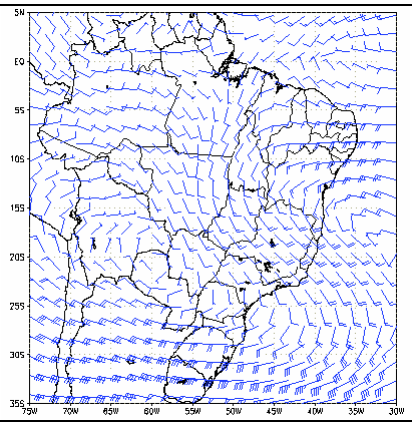
Exemplo 14: Linhas de corrente

```
ga> c
ga> set gxout stream
ga> d uvel;uvel
```



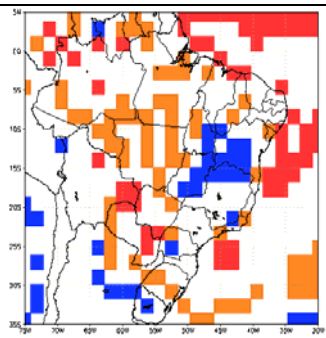
Exemplo 15: Vento com barbela (carta sinótica)

```
ga> c
ga> set gxout barb
ga> d uvel;uvel
```



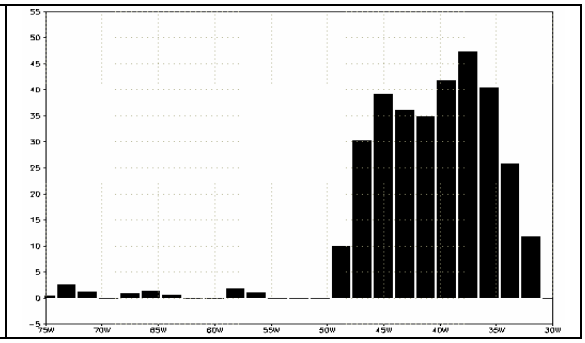
Exemplo 16: Shaded nos pontos de grade dos valores especificados pelo **set fgvals valor cor valor cor ..**

```
ga> c
ga> set gxout fgrid
ga> set fgvals 20 4 23 8 26 2
ga> d temp-273
```



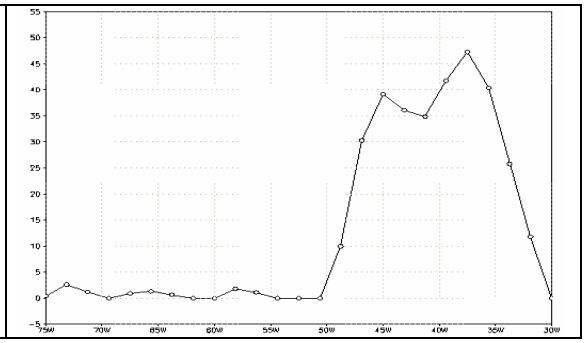
Exemplo 17: Gráfico de barras e gráfico de barra de erros

```
ga> c
ga> set lat 0
ga> set gxout bar
ou
ga> set gxout errbar
ga> d prec
```



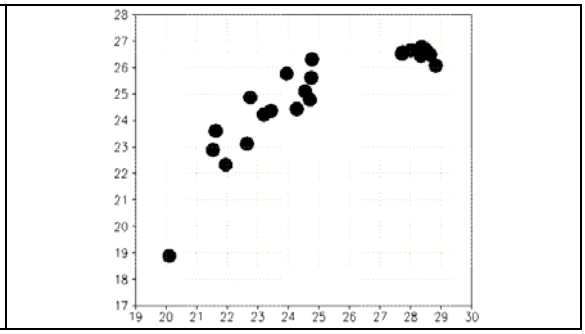
Exemplo 18: Gráfico de linhas

```
ga> c
ga> set gxout line
ga> d prec
```



Exemplo 19: Dispersão

```
ga> c
ga> set gxout scatter
ga> d tsfc-273;temp-273
```



Exemplo 20: Estatística (informações) sobre os dados (sem gráfico)

```
ga> c
ga> set gxout stat
ga> d temp
```

```
ga-> set gxout stat
ga-> d temp-273
Data Type = grid
Dimensions = 0 -1
I Dimension = 41 to 65 Linear -75 1.875
J Dimension = -999 to -999
Sizes = 25 1 25
Undef value = 1e+20
Undef count = 0  Valid count = 25
Min, Max = 18.8816 26.7703
Cmin, cmax, cint = 17 28 1
Stats[sum,sumsqr,root(sumsqr),n]:     628.873 15863.4 125.95 25
Stats[(sum,sumsqr,root(sumsqr))/n]:     25.1229 634.538 25.19
Stats[(sum,sumsqr,root(sumsqr))/(n-1)]: 26.1697 660.977 25.7095
Stats[(sigma,var)(n)]:     1.83756 3.37663
Stats[(sigma,var)(n-1)]:   1.87545 3.51732
ga->
```

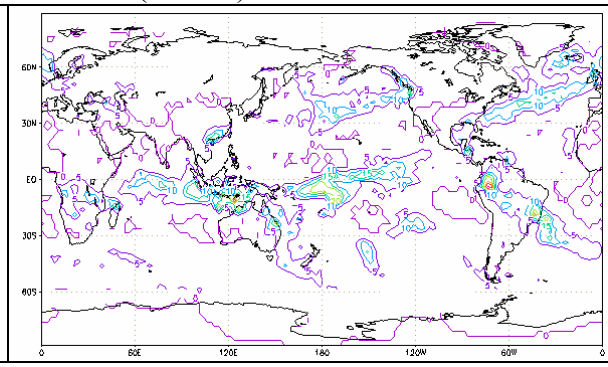
ga> set gxout fwrite	Grava (gera) arquivo grads.fwrite com dados binário (sem gráfico)
ga> set gxout linefill	Linhas com preenchimento de cores entre 2 linhas
ga> set gxout value	Valor da estação (pontos de estações)
ga> set gxout wxsym	Símbolos da Carta Sinótica (condições de tempo)
ga> set gxout findstn	Encontra a estação mais próxima

3.2. Projeções

No GrADS tem-se as opções (tipos) de projeções, conforme exemplos a seguir:

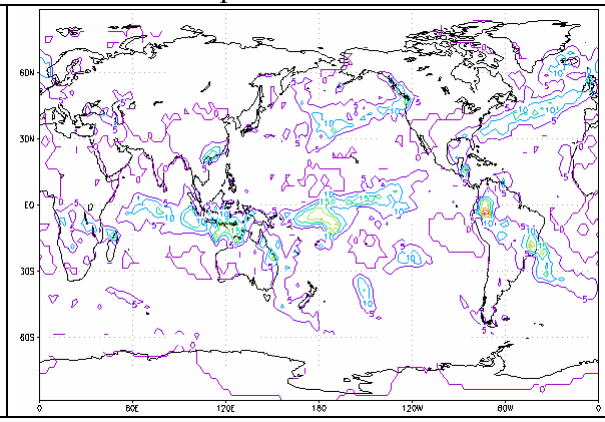
Exemplo 21: Latlon razão de aspecto mantida na tela (default)

```
ga> reinit
ga> open \grads-everaldo\gpcp-1983.ctl
ga> set map 1 1 10
ga> set mproj latlon
ga> d rain
```



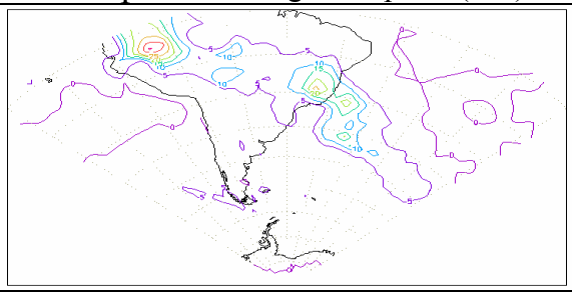
Exemplo 22: Scaled idem a latlon, porém com razão de aspecto não mantida na tela

```
ga> reset
ga> set mproj scaled
ga> d rain
```



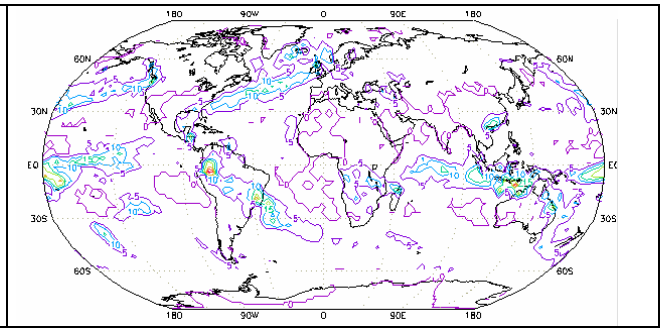
Exemplo 23: sps estereográfica polar (HS) ou nps estereográfica polar (HN)

```
ga> c
ga> set mproj sps
ga> set lon –100 0
ga> set lat –90 0
ga> d rain
```



Exemplo 24: robinson

```
ga> reset
ga> set mproj robinson
ga> set lon –180 180
ga> set lat –90 90
ga> d rain
```

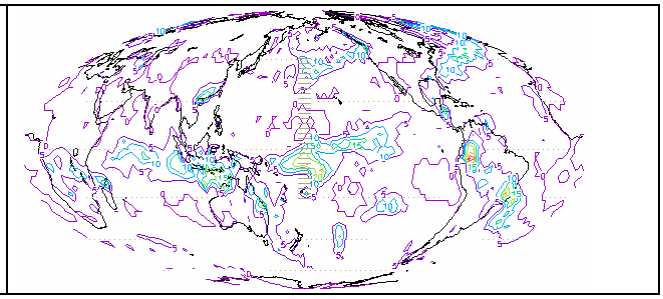


Exemplo 25: orthogr Ortográfica

```
ga> reset
ga> set mproj orthogr
ga> d rain
```

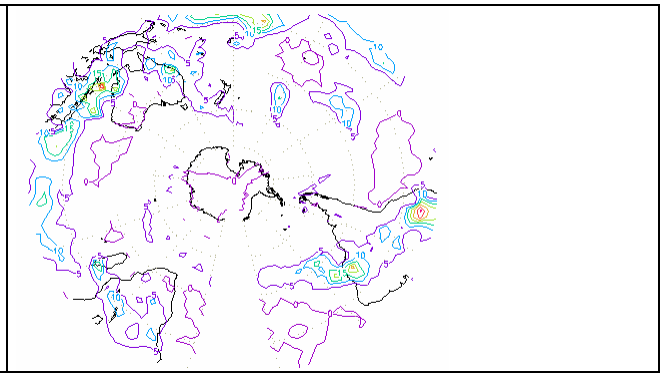
Exemplo 26: mollweide

```
ga> reset
ga> set mproj mollweide
ga> d rain
```



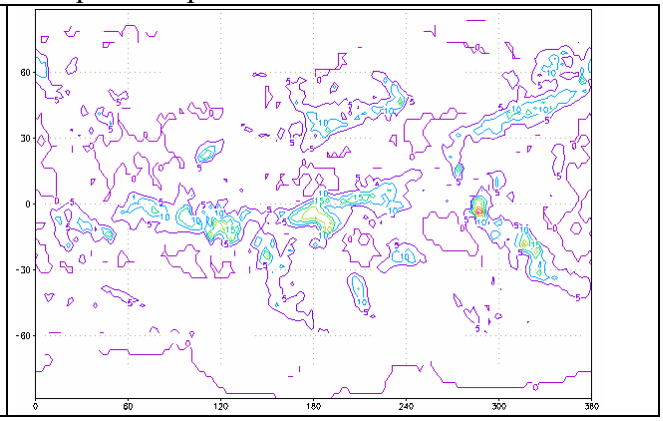
Exemplo 27: lambert Cônica conformal Lambert

```
ga> reset
ga> set mproj lambert
ga> set lat –90 0
ga> d rain
```



Exemplo 28: off idem a scaled, porém não plota mapa e labels sem sinal de lat e lon

```
ga> reset
ga> set mproj off
ga> d rain
```



3.3. Inserindo Títulos, Textos, Formas e Símbolos

As linhas de comando para inserir títulos, textos, formas e símbolos encontram-se abaixo:

ga> draw title Titulo-do-grafico	Escreve título no topo da figura
ga> draw xlab Titulo-X	Escreve título no eixo x
ga> draw ylab Titulo-Y	Escreve título no eixo y
ga> draw string x y Texto	Escreve texto no ponto (x,y)
ga> draw line x1 y1 x2 y2	Desenha uma linha entre (x1,y1) (x2,y2)
ga> draw rec xlo ylo xhi yhi	Desenha um retângulo
ga> draw recf xlo ylo xhi yhi	Desenha um retângulo sólido
ga> draw polyf x1 y1 x2 y2 ... xn yn	Desenha um polígono entre (x1,y1) (x2,y2) ... (xn,yn)
ga> draw mark marktype x y size	Desenha forma no ponto (x,y)
ga> draw wxsym symbol x y size color thickness	Desenha um símbolo de tempo no ponto (x,y)

3.4. Controlando as Opções Gráficas

* Código de cores:

0 = White	8 = orange
1 = black	9 = purple
2 = red	10 = yellow/green
3 = green	11 = med. Blue
4 = blue	12 = dark yellow
5 = cyan	13 = aqua
6 = magenta	14 = dark purple
7 = yellow	15 = grey

Observação: Sequência das cores do arco-iris: 9 14 4 11 5 13 3 10 7 12 8 2 6

Pode usar os comandos:

ga> set ccolor rainbow

ga> set ccolor revrain reverte as cores do arco-iris

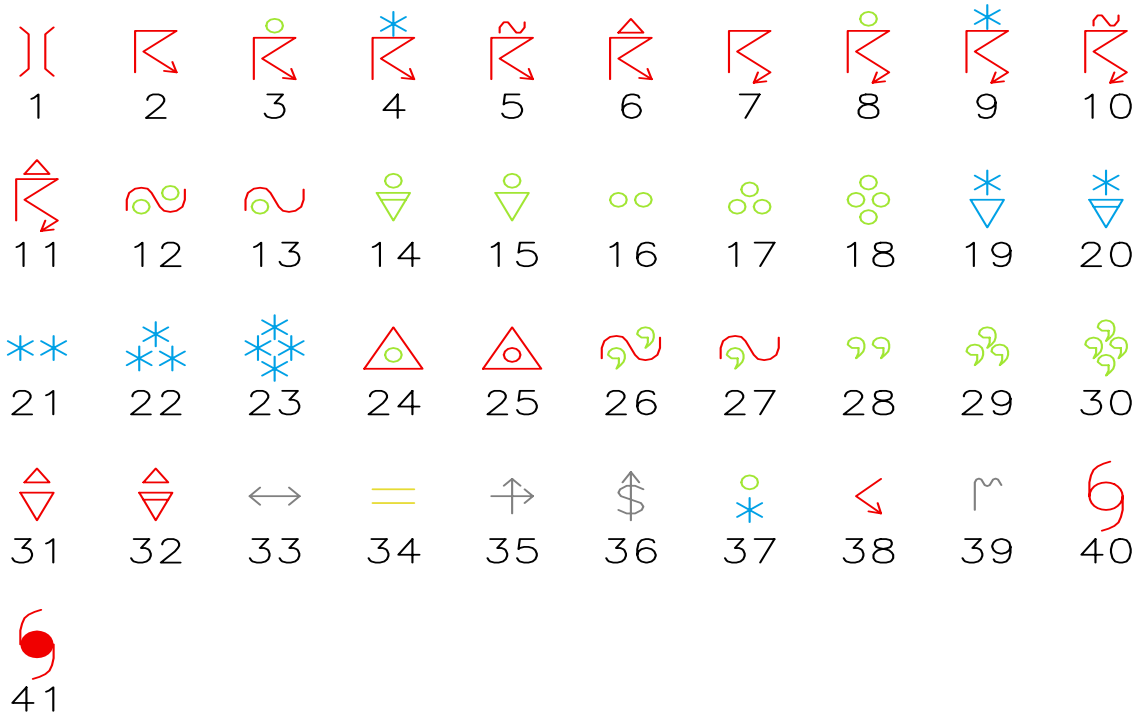
* Código do estilo de linhas:

	0 = none 1 = solid 2 = long dash 3 = short dash 4 = long short dash 5 = dots 6 = dot dash 7 = dot dot dash

* Código de formas (mark):

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

* Código de symbol (de 1 a 41, conforme ilustração abaixo):



* Comando para obter as coordenadas de tela do ponto (x,y):

ga> q pos (Clicar na tela sobre o ponto desejado)

ou

ga> q ll2xy lon lat (Não precisa clicar na tela)

* Comando para controlar texto (string):

ga> set string cor alinhamento espessura rotação

Códigos para alinhamento:

l = left	c = center	r = right
tl = top left	tc = top center	tr = top right
bl = bottom left	bc = bottom center	br = bottom right

ga> set strsiz comprimento altura

ga> set font número tipo da fonte (0 a 5)

* Comandos para controlar as plotagens nos diversos tipos de gráficos:

Gráficos 1-D (gxout = line):

ga> set ccolor código-de-cor	Cor da linha
ga> set cthick valor	Espessura das linhas (1 a 10)
ga> set cstyle código-de-estilo	Estilo da linha
ga> set cmark código-do-marker	Cor do mark
ga> set missconn on | off	Conecta ou não linhas em missing data

Gráficos do tipo (gxout = bar ou errbar):

ga> set bargap valor	Gap entre barras
ga> set barbase valor bottom | top	Plota barras acima ou abaixo do valor
ga> set baropts filled | outline	Barras cheias ou não
ga> set cthick valor	Espessura das linhas (1 a 10)

Gráficos do tipo (gxout = linefill):

ga> set lfcols cor1 cor2	cores 1 e 2 entre as isolinhas

Gráficos do tipo (gxout = contour):

ga> set ccolor código-de-cor	Cor da isolinha
ga> set cthick valor	Espessura das isolinhas (1 a 10)
ga> set cstyle código-de-estilo	Estilo da isolinha
ga> set cterp on | off	Aplica ou não suavização
ga> set cint valor	Intervalo entre as isolinhas
ga> set cmax valor	Controla o valor Maximo das isolinhas
ga> set cmin valor	Controla o valor Mínimo das isolinhas

ga> set black valor1 valor2	Contornos omitidos entre valor1 e valor2
ga> set clevs valor1 valor2 ...	Plota valores especificados
ga> set ccols cor1 cor2 ...	Especifica cores para clevs
ga> set rbrange valor1 valor2	valor1 e valor 2 para rainbow
ga> set rbcols cor1 cor2 ...	Especifica cores para clevs
ga> set rbcols auto	Cores em rainbow
ga> set clab on | off | forced	Mostra ou não os valores das isolinhas
ga> set clskip valor-do-intervalo	valores das isolinhas em intervalos de
ga> set clopts cor estilo tamanho	Especifica cor, estilo e tamnaho do label
ga> set csmooth on | off	Aplica suavização

Gráficos do tipo (gxout = shaded or grfill):

ga> set cint valor	Intervalo entre as isolinhas
ga> set cmax valor	Controla o valor Maximo das isolinhas
ga> set cmin valor	Controla o valor Mínimo das isolinhas
ga> set black valor1 valor2	Contornos omitidos entre valor1 e valor2
ga> set clevs valor1 valor2 ...	Plota valores especificados
ga> set ccols cor1 cor2 ...	Especifica cores para clevs
ga> set rbrange valor1 valor2	valor1 e valor 2 para rainbow
ga> set rbcols cor1 cor2 ...	Especifica cores para clevs
ga> set csmooth on | off	Aplica suavização

Gráficos do tipo (gxout = grid):

ga> set dignumber numero	número dígitos depois da casa decimal
ga> set digsize numero	tamanho dos números

Gráficos do tipo (gxout = vector ou barb):

ga> set ccolor código-de-cor	Cor dos vetores
ga> set cthick valor	Espessura dos vetores (1 a 10)
ga> set arrlab on | off	mostra ou não vetor de referência abaixo do plot
ga> set arrscl valor magnitude	comprimento do vetor de acordo com magnitude
ga> set arrowhead valor	tamanho da cabeça da seta
ga> set cint valor	Intervalo
ga> set cmax valor	Controla o valor Maximo
ga> set cmin valor	Controla o valor Mínimo
ga> set black valor1 valor2	Não plota vetores entre valor1 e valor2
ga> set clevs valor1 valor2 ...	Plota valores especificados
ga> set ccols cor1 cor2 ...	Especifica cores para clevs
ga> set rbrange valor1 valor2	valor1 e valor 2 para rainbow
ga> set rbcols cor1 cor2 ...	Especifica cores para clevs

Gráficos do tipo (gxout = scatter):

ga> set cmark código-do-marker	Cor do mark

ga> set digsize numero tamanho dos números
ga> set ccolor código-de-cor Especifica cor
ga> set vrange valor1 valor2 range entre valor1 e valor2 do eixo y
ga> set vrange2 valor1 valor2 range entre valor1 e valor2 do eixo x

Gráficos do tipo (gxout = fgrid):
ga> set fgvals valor cor valor cor .. especifica valores e cores para fgrid

Gráficos do tipo (gxout = stream):
ga> set strmden valor densidade das linhas de corrente
ga> set ccolor código-de-cor Cor da isolinha
ga> set cint valor Intervalo entre as isolinhas
ga> set cmax valor Controla o valor Maximo das isolinhas
ga> set cmin valor Controla o valor Mínimo das isolinhas
ga> set cthick valor Espessura das isolinhas (1 a 10)
ga> set black valor1 valor2 Contornos omitidos entre valor1 e valor2
ga> set clevs valor1 valor2 ... Plota valores especificados
ga> set ccols cor1 cor2 ... Especifica cores para clevs
ga> set rbrange valor1 valor2 valor1 e valor 2 para rainbow
ga> set rbcols cor1 cor2 ... Especifica cores para clevs

Dados de estações; Gráfico do tipo (gxout = value):
ga> set digsize numero tamanho dos números
ga> set ccolor código-de-cor Especifica cor
ga> set stid on | off2 mostra ou não id da estação
ga> set cthick valor Espessura (1 a 10)

Dados de estações; Gráfico do tipo (gxout = barb):
ga> set digsize numero tamanho dos números
ga> set ccolor código-de-cor Especifica cor
ga> set cthick valor Espessura (1 a 10)

Dados de estações; Gráfico do tipo (gxout = wxsym):
ga> set ccolor código-de-cor Especifica cor
ga> set cthick valor Espessura (1 a 10)
ga> set digsize numero tamanho dos números
ga> set wxcols cor1 cor2 ... Especifica cores para symbols

Dados de estações; Gráfico do tipo (gxout = model):
ga> set ccolor código-de-cor Especifica cor
ga> set cthick valor Espessura (1 a 10)
ga> set digsize numero tamanho dos números
ga> set wxcols cor1 cor2 ... Especifica cores para symbols
ga> set mdlopts noblank | blank | dig3 | nodig3 opções de model

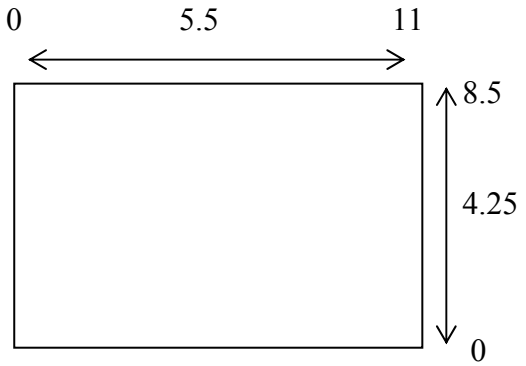
* Comandos para controlar eixos, mapas, etc:

ga> set grid on | off | horizontal | vertical	Linhas de grade conforme opções
ga> set zlog on | off	Escala vertical logarítmica ou não
ga> set xaxis inicio fim incremento	range eixo x do inicio ao fim com incremento
ga> set yaxis inicio fim incremento	range eixo y do inicio ao fim com incremento
ga> set xlevs label1 label2 ...	labels específicos para eixo x
ga> set ylevs label1 label2 ...	labels específicos para eixo y
ga> set xlint intervalo	intervalo para eixo x
ga> set ylint intervalo	intervalo para eixo y
ga> set xyrev on	inverte os eixos
ga> set xflip on	inverte a ordem do eixo x
ga> set yflip on	inverte a ordem do eixo y
ga> set xlopts cor espessura tamanho	cor espessura e tamanho do label eixo x
ga> set ylopts cor espessura tamanho	cor espessura e tamanho do label eixo y
ga> set annot cor espessura	cor e espessura do string (draw title, etc)
ga> set mpdset lowres | mres | hires | brmap	resolução dos mapas
ga> set map cor estilo espessura	cor, estilo e espessura da linha do mapa
ga> set mpdraw on | off	plota ou não mapas
ga> set grads on | off	coloca/tira logotipo do GrADS

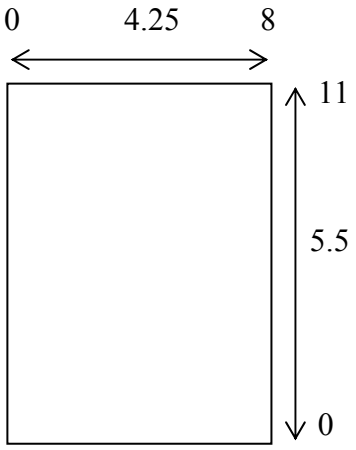
3.5. Controle de Página

Tamanhos padrões da tela de visualização:

grads –l (landscape: 11 x 8.5)



grads –p (portrait: 8.5 x 11)



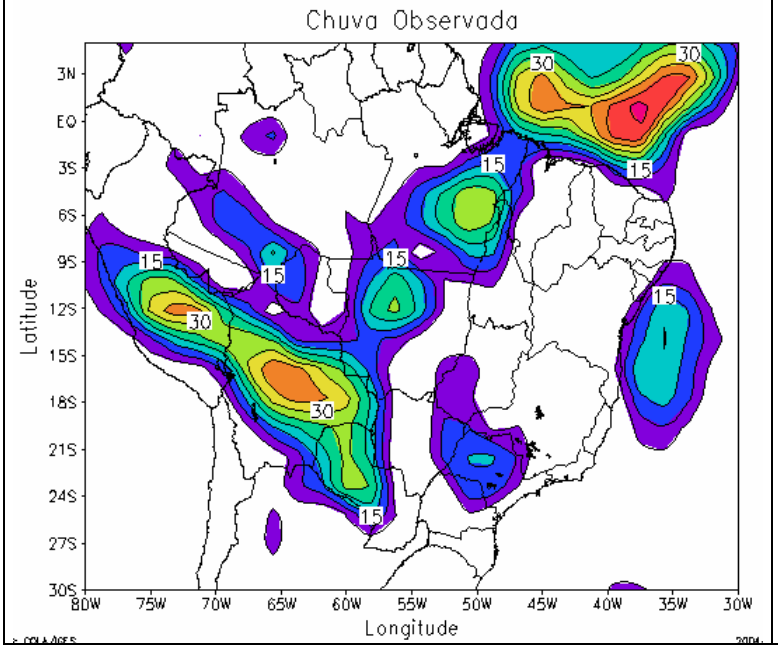
* Comandos de controle de página:

Virtual Page	Print Área
ga> set vpage xmin xmax ymin ymax	**ga> set parea xmin xmax ymin ymax**
ga> set vpage off	**ga> set parea off**

3.6. Exemplos e Exercícios

Exemplo 29:
Mapa de Chuva

```
ga> reinit
ga> open exemplo.ctl
ga> set display color white
ga> c
ga> set mpdset hires brmap
ga> set map 1 1 10
ga> set grid off
ga> set xlopts 1 1 0.15
ga> set ylopts 1 1 0.15
ga> set lat -30 5
ga> set lon –80 –30
ga> set gxout shaded
ga> set cmin 1
ga> set cint 5
ga> d prec
ga> set gxout contour
ga> set cmin 1
ga> set cint 5
ga> set ccolor 1
ga> set clab on
ga> set clskip 3
ga> d prec
ga> draw title Chuva Observada
ga> draw xlab Longitude
ga> draw ylab Latitude
```

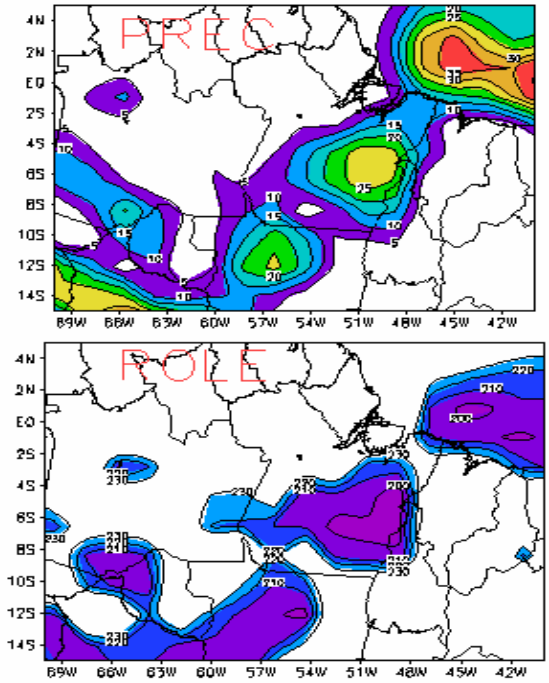


Exercício proposto 9:

Plotar campo de pressão ao nível do mar
Destacando em shaded somente as altas pressões (psnm > 1015)
E vetor vento em barbela (lembre de dar o skip) sobre toda a grade da América do Sul
Coloque os títulos
Escreva os string de A e B nos centros de aixa e alta

Exemplo 30:
Duas figuras na mesma página portrait
Chuva e ROLE sobre a Amazônia

```
ga> set mpdset hires brmap
ga> set map 1 1 10
ga> set grid off
ga> set grads off
ga> set xlopts 1 1 0.15
ga> set ylopts 1 1 0.15
ga> set lat -15 5
ga> set lon –70 –40
ga> set parea 0.5 8 6 10.8
ga> set gxout shaded
ga> set cmin 1
ga> set cint 5
ga> d prec
ga> set gxout contour
ga> set cmin 1
ga> set cint 5
ga> set ccolor 1
ga> d prec
ga> set parea 0.5 8 0.5 5.5
ga> set gxout shaded
ga> set cmax 230
ga> set cint 10
ga> d role
ga> set gxout contour
ga> set cmax 230
ga> set cint 10
ga> set ccolor 1
ga> d role
```



Exercício 10:
Plote 4 figuras usando a opcao ***vpage*** *na mesma página landscape*

Variáveis:

Vetor vento em 850 hPa

Linhas de corrente em 200 hPa

Temperatura na superfície

Geopotencial em 500 hPa

Coloque os títulos em cada figura

4. GERANDO ARQUIVOS DE SAÍDAS GRÁFICAS

4.1. Arquivo GrADS metafile (.gmf)

* Gerando um arquivo GrADS metafile (*.gmf)

O exemplo abaixo plota o campo de temperatura e gera o arquivo .gmf

Exemplo 31: procedimento para gerar um .gmf

ga> enable print arquivo1.gmf	abre o arquivo
ga> d temp	
ga> print	grava/salva o arquivo no disco
ga> disable print	fecha o arquivo

Observações:

- Se o usuário não fizer o **disable print**; o arquivo também é finalizado com **reinit** ou **quit**
- É possível gerar vários gráficos (frames) separados dentro de um mesmo .gmf

4.2. GrADS Metafile Viewer for Windows

O GrADS metafile Viewer (GV) é um aplicativo em ambiente windows que serve para fazer a visualização e manipulação dos arquivos .gmf gerados no GrADS.

Os gráficos abertos dentro do GV podem ser copiados e colados nos seus documentos (Word, PowerPoint, etc). Há também outras opções, tais como: impressão, recorte de um pedaço da figura, etc.

4.3. Aplicativo gxtran

O aplicativo utilitário **gxtran** é utilizado para manipular e visualizar arquivos .gmf. É mais utilizado em ambiente UNIX, conforme a sintaxe abaixo:

ga> ! gxtran opções -i arquivo.gmf

As opções são:

-a	anima os frames sem dar o enter em cada troca de frame
-r	reverte cores de fundo
-g comprimento x altura	geometria da janela (tamanho em pixel)

Observação: tecle <enter> para sair do gxtran

Exemplo 32: Gerando um .gmf e visualizando com o gxtran

```
ga> c
ga> enable print arquivo2.gmf
ga> d temp(z=1)
ga> print
ga> c
ga> d temp(z=3)
ga> print
ga> c
ga> d temp(z=5)
ga> print
ga> c
ga> d temp(z=7)
ga> print
ga> disable print

ga> ! gxtran –a –g 800x600 –i arquivo2.gmf
```

Use também o GV e veja que a facilidade de manipulação (windows) é melhor...

4.4. Aplicativos gxps e gxeps

O aplicativo utilitário **gxps** (versões para ambientes windows e Unix) converte arquivos .gmf em imagens no formato PostScript (.ps), conforme a sintaxe abaixo

ga> ! gxps opções –i arquivo.gmf –o arquivo.ps

As opções são:

-c	formato colorido
-r	fundo preto
-d	coloca CTRL-D no final do arquivo (impressão HP)

O aplicativo utilitário **gxeps** (versões para ambientes windows e Unix) também converte arquivos .gmf em imagens no formato PostScript (.eps), com opções adicionais, conforme a sintaxe abaixo

ga> ! gxeps opções –i arquivo.gmf –o arquivo.eps

As opções são:

-c	formato colorido
-r	fundo preto
-d	coloca CTRL-D no final do arquivo (impressão HP)
-1	PostScript Level 1
-2	PostScript Level 2
-a	Página tamanho A4
-l	Página tamanho Carta

-L	Prompt para um label a ser colocado na figura
-n	Prompt para uma nota a ser incluída no arquivo
-v	modo verbose

Observação: Em ambos **gxps** e **gxeps**, se não especificar –c a imagem será em escala de cinza no fundo branco.

4.5. Comandos printim e wi

O comando **printim** (versões para ambientes windows e Unix) converte o conteúdo gráfico da janela em um arquivo do tipo imagem (GIF ou PNG), conforme sintaxe abaixo:

ga> printim arquivo.out opções

As opções são:

gif	gera imagem do tipo GIF (default: imagem PNG)
black	fundo preto
white	fundo branco
xNNN	tamanho em pixel horizontal
yNNN	tamanho em pixel vertical

O comando **wi** (versões para ambientes windows e Unix) usa a interface do ImageMagick library converte o conteúdo gráfico da janela em um arquivo do tipo imagem (vários formato), conforme sintaxe abaixo:

ga> wi arquivo.out

As opções de formato do ImageMagick a serem escolhidas na extensão .out são:
gif, bmp, cgm, eps, fax, ico, jpeg, pcx, hdf e outras...

Observações:

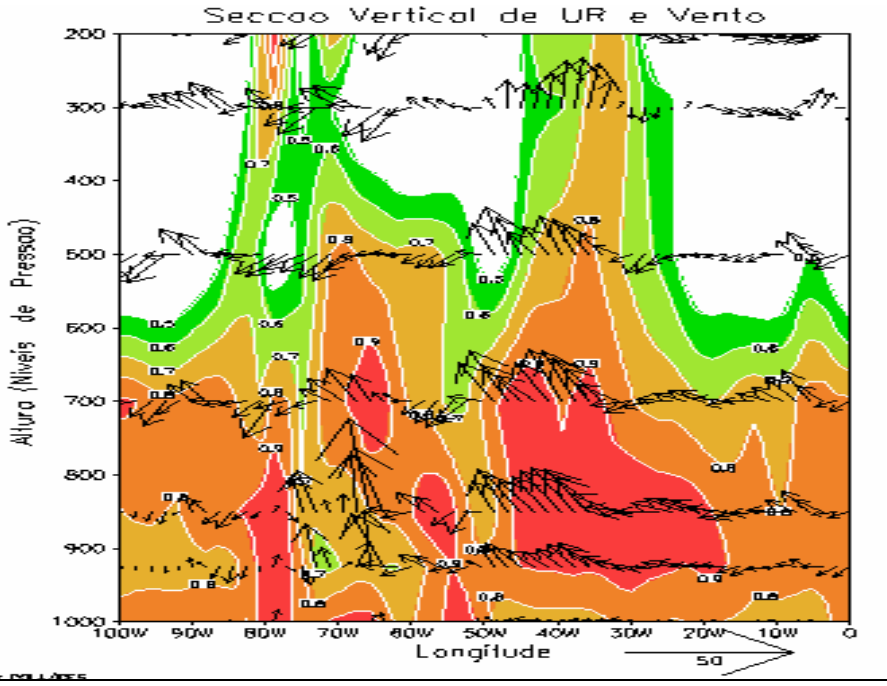
- O **printim** também funciona no modo batch, porém só no GrADS versão 1.8 ou superior
- O **wi** não roda no modo batch, pois requer um X-server. Alguns formatos do ImageMagick (TIFF, PNG, MPEG, etc) não funcionam no GrADS. Nesse caso, a imagem gerada será do tipo MIFF. Se nenhuma extensão for especificada, GIF é o formato default.

4.6. Exemplos e Exercícios

Exemplo 33 – Secção vertical (Longitude x Altura) de UR e Vento (Uvel;Omega) com geração do .gmf a ser colocado no Word como figura

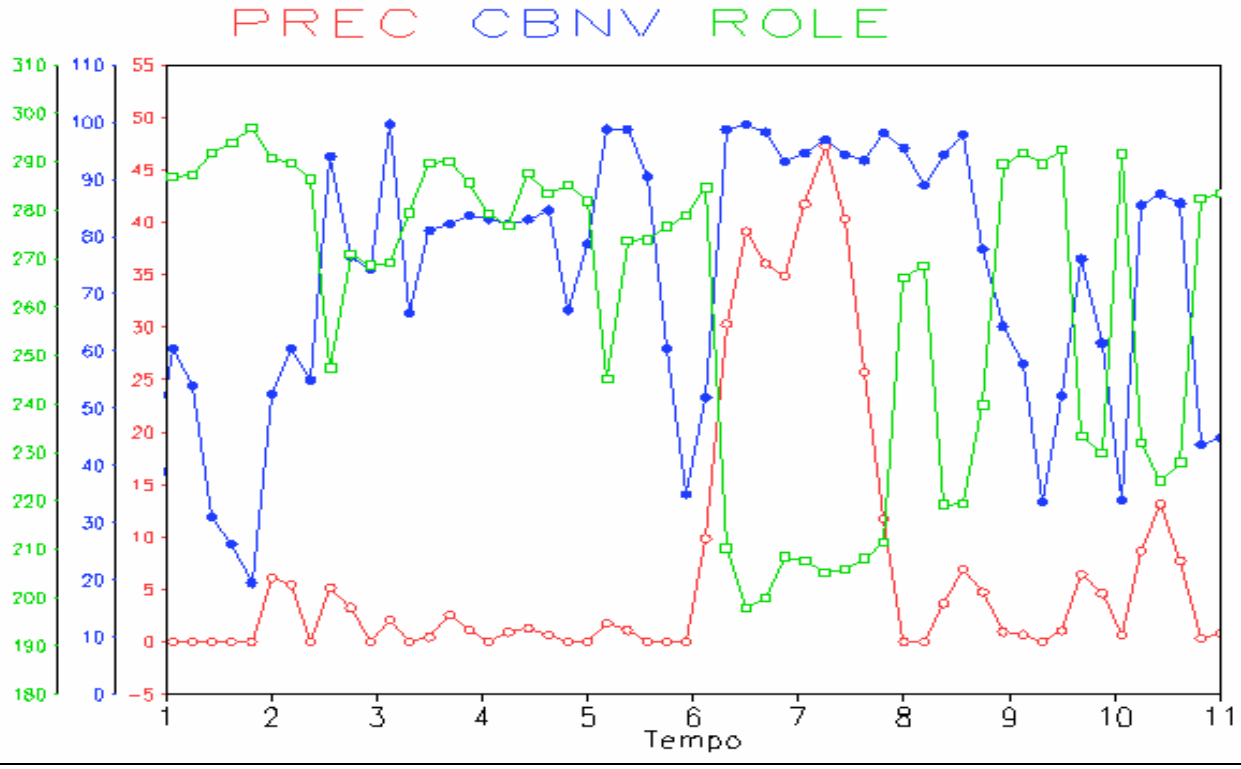
```
ga> open exemplo.ctl
ga> set lon -100 0
ga> set lat 0
ga> set z 1 7
ga> enable print ex33.gmf
ga> set gxout shaded
ga> set cmin 0.5
ga> set cint 0.1
ga> d umrl
ga> set gxout contour
ga> set ccolor 0
ga> set cmin 0.5
ga> set cint 0.1
ga> d umrl
ga> set gxout vector
ga> set ccolor 1
ga> set arrscl 1.5 50
ga> set arrowhead -0.5
ga> set cthick 10
ga> d uvel;omeg*(–100)
ga> draw title Seccao Vertical de UR e Vento
ga> draw xlab Longitude
ga> draw ylab Altura (Niveis de Pressao)
ga> print
ga> disable print
```

Depois de gerar o ex33.gmf, abra-o no GV e coloque (copy; paste) no seu Word como figura



Exemplo 34 – Gráfico de linhas com geração do .gmf a ser colocado no Word como figura

```
ga> enable print ex34.gmf
ga> set parea 2 10.8 1 7.7
ga> set lon -100 0
ga> set lat 0
ga> set grid off
ga> set grads off
ga> set xaxis 1 11 1
ga> set xlopts 1 1 0.2
ga> set gxout line
ga> set ccolor 2
ga> set ylopts 2 1 0.12
ga> d prec
ga> set ccolor 4
ga> set ylopts 4 1 0.12
ga> d cbnv*100
ga> set ccolor 3
ga> set ylopts 3 1 0.12
ga> d role
ga> set strsiz 0.4 0.3
ga> set string 2
ga> draw string 2.5 8 PREC
ga> set string 4
ga> draw string 4.5 8 CBNV
ga> set string 3
ga> draw string 6.5 8 ROLE
ga> draw xlab Tempo
ga> print
ga> disable print
```



5. VARIÁVEIS, EXPRESSÕES E FUNÇÕES

5.1. Nomes das Variáveis

A especificação completa para um nome de variável é:

abbrev.file#(dimexpr,dimexpr,...)

abbrev	abreviação para a variável especificada no CTL
file#	número do arquivo aberto que contém a variável O default é 1 (primeiro arquivo aberto). O comando **set dfile file#** muda de arquivo.
dimexpr	expressão da dimensão que modifica localmente o ambiente da dimensão corrente somente para a variável em questão. Somente dimensões fixas podem ser usadas.

As dimensões absolutas são:

X | Y | Z | T | LON | LAT | LEV | TIME = valor

As dimensões relativas são, por exemplo:

X | Y | Z | T | LON | LAT | LEV | TIME + – / valor

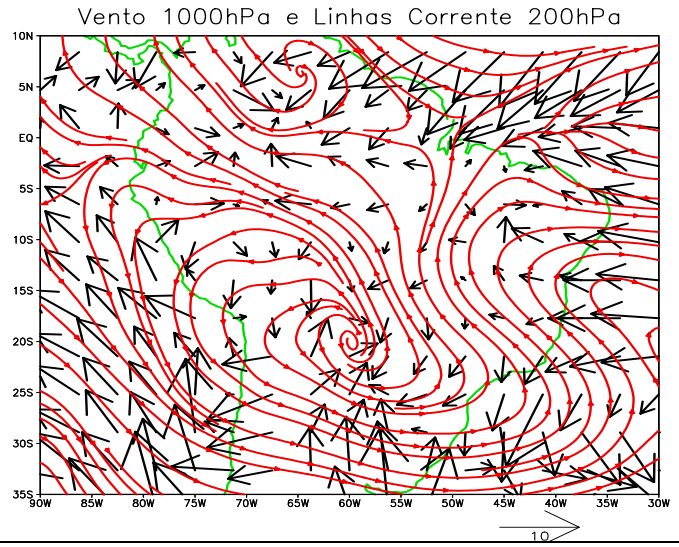
Alguns exemplos de especificações de variáveis:

zgeo.3(lev=500)	arquivo 3, expressão de dimensão absoluta
prec(time-12hr)	expressão de dimensão relativa
uvel.2(t-1,lev=850)	expressão com duas dimensões

Observação: lat, lon, lev são variáveis pré-definidas pelo GrADS, isto é, estão implicitamente contidas dentro de cada CTL. Quando usadas, fornecem a lat, lon, lev nos respectivos pontos de grade, por exemplo: lat.2 especifica as latitudes da grade do segundo CTL aberto.

Exemplo 35 – Usando expressões...

```
ga> set map 3 1 10
ga> set lon -90 -30
ga> set lat -35 10
ga> set lev 1000
ga> set cthick 10
ga> set arrscl 1 10
ga> set arrowhead -0.5
ga> d skip(uvel,2);vvel
ga> set gxout stream
ga> set ccolor 2
ga> set strmden 2
ga> d uvel(lev=200);vvel(lev=200)
```



5.2. Definindo Novas Variáveis: define

O comando **define** permite a criação interativa de novas variáveis, conforme a sintaxe:

define nome-da-variável = expressão

A nova variável é armazenada na memória e pode ser usada em comandos subseqüentes. É possível fazer o define com dimensões variando de 0 a 4. Quando Z e/ou T estão variando, o define avalia a expressão para cada Z e T.

Para limpar a memória e undefining a sua nova variável use o comando **undefine**, conforme a sintaxe:

undefine nome-da-variável

Exemplo 36 – Definindo uma variável para vários níveis verticais

```
ga> set lon -90 -30
ga> set lat -35 10
ga> set lev 1000 200

ga> define tempc = temp + 273

ga> set lev 1000
ga> d tempc

ga> set lev 500
ga> d tempc
```

5.3. Expressões

As expressões no GrADS consistem de operadores, operandos e parênteses, os quais são usados de maneira similar ao FORTRAN para controlar a ordem dos cálculos nas operações.

Os operadores são: + (adição), – (subtração), * (multiplicação), / (divisão)

Os operandos podem ser: especificações de variáveis, funções e constantes

Observação: As operações são realizadas para cada ponto de grade e, por isso, as grades devem possuir as mesmas dimensões.

Exemplos:

zgeo – zgeo(t–1)
temp(lev=500) – temp(lev=850)

5.4. Funções

O GrADS possui uma grande variedade de funções intrínsecas. A seguir enumera-se a listagem de acordo com atribuições específicas, bem como a sintaxe de cada uma delas.

* Operações matemáticas:

abs(expr) fornece o valor absoluto de expr
(dados em pontos de grade e pontos de estações)

cdiff(expr,dim) operação de diferença centrada em expr na direção especificada por dim
Valores nas bordas da grade são missing

Exemplo: Cálculo da advecção de temperatura:

```
define dtx = cdiff(temp,x)
define dty = cdiff(temp,y)
define dx = cdiff(lon,x)*3.1416/180
define dy = cdiff(lat,y)*3.1416/180
d -1*( (uvel*dtx)/(cos(lat*3.1416/180)*dx) + vvel*dty/dy )/6.37e6
```

exp(expr) cálculo do exponencial de expr (operação: e^x, onde x é a expr)
(dados em pontos de grade e pontos de estações)

gint (expr) integral de expr (similar a ave, mas não divide pela área total)

log(expr) cálculo do logaritmo natural de expr
Valores menores ou igual a zero são missing.
(dados em pontos de grade e pontos de estações)

log10(expr) idem acima, porém para o logaritmo na base 10

pow(expr1,expr2) eleva valor expr1 na potência expr2 (operação: x^y, x=expr1; y=expr2)
(dados em pontos de grade e pontos de estações)

sqrt(expr) raiz quadrada de expr. Valores menores do que zero são missing
(dados em pontos de grade e pontos de estações)

vint(psexpr,expr,top) integral vertical de expr com mass-weighted

psexpr	expressão para pressão na superfície em mb ou hPa (equivale ao limite da integral na superfície)
expr	variável a ser integrada (variando somente em X e Y)
top	pressão no topo (equivale ao limite da integral no topo). É uma constante e não pode ser uma expressão

Exemplo: cálculo da água precipitável em mm

```
vint(psnm,umes,275)
```

* Funções trigonométricas:

cos(expr)	coseno de expr em radianos. Vale para pontos de grade e estações
acos(expr)	coseno inverso de expr em radianos. Valores de expr maior do que 1 e menor do –1 são missing
sin(expr)	seno de expr em radianos. Resultado entre –1 e 1 Vale para pontos de grade e estações
asin(expr)	idem a acos(expr), mas para o seno inverso
tan(expr)	tangente de expr em radiano. Vale para pontos de grade e estações
atan2 (expr1, expr2)	tangente inversa de expr1/expr2 em radianos

* Médias e somatórios:

aave(expr, xdim1, xdim2, ydim1, ydim2) média espacial de expr na grade X,Y considera latitude-weighted

expr	expressão da variável
xdim1	expressão da dimensão X (ponto inicial)
xdim2	expressão da dimensão X (ponto final)
ydim1	expressão da dimensão Y (ponto inicial)
ydim2	expressão da dimensão Y (ponto final)

Exemplo: Numa grade global seria
aave(expr, lon=0, lon=360, lat=-90, lat=90)
ou
aave(expr, global) ou **aave(expr, g)**

amean (expr, xdim1, xdim2, ydim1, ydim2) idem a aave, mas not latitude-weighted

asum(expr, xdim1, xdim2, ydim1, ydim2) somatório de expr na grade X,Y considera grid-weighted

asumg(expr, xdim1, xdim2, ydim1, ydim2) idem a asum, mas not grid-weighted

ave(expr, dim1, dim2 <,tinc> <,-b>) média de expr na dimensão especificada considera grid weighted

expr	expressão da variável
dim1	ponto inicial da dimensão
dim2	ponto final da dimensão
tinc	incremento opcional para o caso de média na dimensão T
-b	usa limits exatos

Exemplo:
Media zonal global de temperatura: **ave(temp,lon=0,lon=360)**
Desvio padrão de chuva anual (série 30 anos): **define cli = ave(prec,t=1,t=30)**
sqrt(ave(pow(cli-prec,2),t=1,t=30))

mean (expr, dim1, dim2, <,tinc> <,-b>) idem a ave, mas not grid weighted

sum (expr, dim1, dim2, <,tinc> <,-b>) somatório de expr na dimensão especificada considera grid weighted

sumg (expr, dim1, dim2, <,tinc> <,-b>) idem a sum, mas not grid weighted

tmave(maskexpr,expr,timexpr1,timexpr2) media temporal quando da aplicação da expressão de máscara

maskexpr	expressão de máscara
expr	expressão de variável
timexpr1,2	limites da dimensão temporal

* Correlação e regressão:

scorr(expr1, expr2, xdim1, xdim2, ydim1, ydim2) correlação espacial entre duas variáveis na dimensão X,Y

expr1	expressão da variável 1
expr2	expressão da variável 2
xdim1	expressão da dimensão X (ponto inicial)
xdim2	expressão da dimensão X (ponto final)
ydim1	expressão da dimensão Y (ponto inicial)
ydim2	expressão da dimensão Y (ponto final)

Exemplo: Correlação entre a precipitação anual e radiação de onda longa sobre o Brasil (resultado: um valor entre –1 e 1)

```
set lat -35 5
set lon –80 -30
d scorr(prec, role, lon=-80, lon=-30, lat=-35, lat=5)
```

tcorr (expr1, expr2, tdim1, tdim2) mapa de correlação espacial entre a série temporal de expr1 e expr2 numa grade X,Y

expr1	série temporal da variável 1
expr2	variável 2 na dimensão X,Y
tdim1	expressão de tempo inicial
tdim1	expressão de tempo final

Exemplo: Correlação entre série de 30 anos da chuva anual em Belém e a radiação de onda longa sobre o Brasil tropical

```
set lat –1.5
set lon -48
set z 1
set t 1 30
define belem = prec
set lon –80  -30
set lat –15  5
```

```
set z 1
set t 1
d tcorr(belem, role, t=1, t=30)
```

sregr(expr1, expr2, xdim1, xdim2, ydim1, ydim2) regressão linear (mínimos quadrados) entre expr1 e expr2 numa grade X,Y

tregr (expr1, expr2, tdim1, tdim2) regressão dos mínimos quadados entre expr1 e expr2 dependentes do tempo

* Variáveis meteorológicas derivadas e operações vetoriais

tvrh2q(tvexpr,rhexpr) cálculo de umidade específica q em g/g a partir da temperatura virtual e umidade relativa

tvexpr temperatura virtual em Kelvin
rhexpr umidade relativa em percentagem (0 a 100)

tvrh2t(tvexpr,rhexpr) cálculo de temperatura em Kelvin a partir da temperatura virtual e umidade relativa

hcurl(uexpr,vexpr) cálculo da componente vertical da vorticidade

uexpr e vexpr são as componentes zonal e meridional do vento, respectivamente

hdivg(uexpr,vexpr) cálculo da divergência horizontal por diferenças finitas

mag(uexpr,vexpr) cálculo da magnitude do vento horizontal (wind speed)

skip (expr, skipx, skipy) seta valores alternantes de expr na grade X,Y. Usada principalmente para diminuir a densidade de vectors e barbs

expr expressão da variável
skipx fator de skip na direção X
skipy fator de skip na direção Y

* Operações em ponto de grade:

fndlvl (expr, expr_to_find, lev1, lev2) Dados 2 variáveis (expr e expr_to_find) na dimensão X,Y, acha-se o primeiro nível vertical em que o valor de expr_to_find ocorre em expr. lev1 e lev2 especificam o range de níveis sobre a ser procurado.

Exemplo: Encontrar os níveis de pressão da isoterma de 30 graus entre 1000 e 200 hPa

```
d fndlvl (temp, const(temp,30), lev=1000, lev=200)
```

max(expr, dim1, dim2 <,tinc>) máximo valor de expr na dimensão especificada
tinc é opcional no caso da dimensão ser T

maxloc(expr, dim1, dim2 <,tinc>) fornece a coordenada do máximo valor de expr na dimensão especificada

min(expr, dim1, dim2 <,tinc>) mínimo valor de expr na dimensão especificada
tinc é opcional no caso da dimensão ser T

minloc(expr, dim1, dim2 <,tinc>) fornece coordenada do mínimo valor de expr na dimensão especificada

smth9(expr) suavização de 9-pontos na expr

* Outras:

const (expr, value, <-u|-a>) mudança de valores missing para uma constante;
mudança de valores non-missing de uma variável para uma constante

expr	variável
value	constante inteira ou ponto flutuante
-u	usa todos dados missing; dados non-missing não são mudados
-a	todos dados são mudados

Exemplo: plotando uma linha horizontal no gráfico linefill

```
set lon 0
set lat -35 10
set gxout linefill
set lev 1000
d const((temp-273), -20);temp-273
```

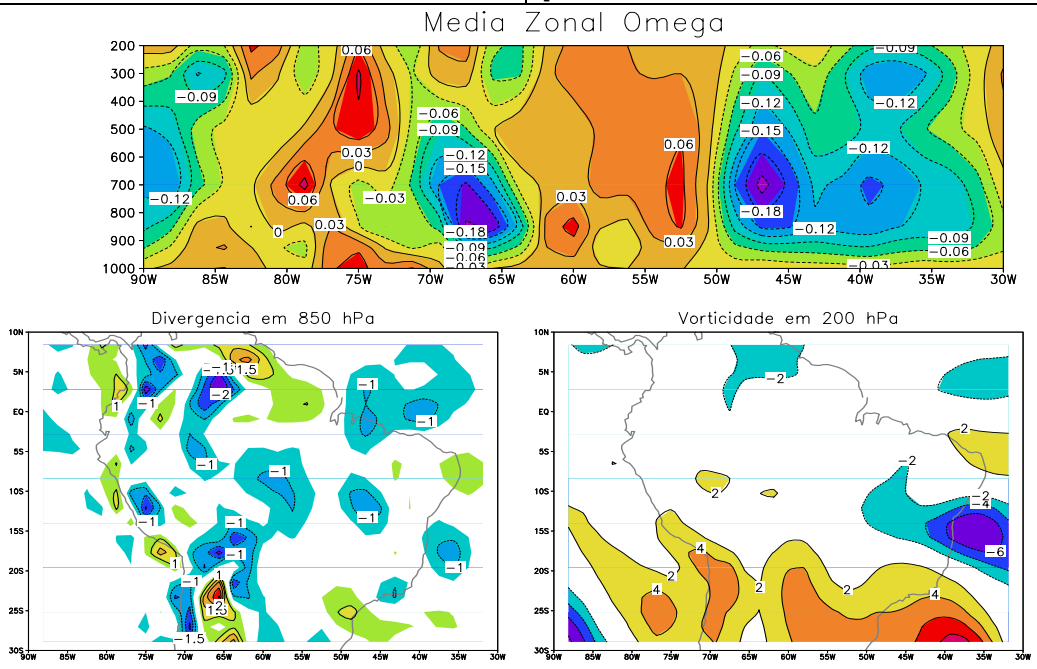
maskout(expr,mask) valores de mask ou menor do que zero não são plotados

5.5. Exemplos e Exercícios

Exemplo 37 – Usando funções para o cálculo de variáveis derivadas (escrever exemplo37.gs)

```
'open exemplo.ctl'

'enable print ex37.gmf'

'set   lon -90 -30'
'set   lat -30 10'
'set   lev  1000 200'
'define medz = ave(omeg, lat=-5, lat=5)'

'set vpage 0 11 4.25 8.5'
'set lat 0'
'set gxout shaded'
'd medz'
'set gxout contour'
'd medz'
'draw title Media Zonal Omega'
'set vpage off'

'set   lon -90 -30'
'set   lat -30 10'
'set   lev  200'
'define vort = hcurl(uvel,vvel)'
'set   lev  850'
'define dive = hdivg(uvel,vvel)'

'set map 15 1 10'
```

```
'set vpage 0 5.5 0 5'
'set clopts 1  1  .15'
'set grads off'
'set grid off'
'set gxout shaded'
'set black -.5 .5'
'd dive/1e-5'
'set gxout contour'
'set black -.5 .5'
'd dive/1e-5'
'draw title Divergencia em 850 hPa'
'set vpage off'

'set vpage 5.5 11 0 5'
'set grads off'
'set grid off'
'set gxout shaded'
'set black -.5 .5'
'd vort/1e-5'
'set gxout contour'
'set black -.5 .5'
'd vort/1e-5'
'draw title Vorticidade em 200 hPa'
'set vpage off'

'enable print ex35.gmf'
'print'
```



6. LINGUAGEM DE PROGRAMAÇÃO (script.gs)

6.1. Conceitos Básicos

O GrADS possui uma interface programável (scripting language) em que, basicamente, o usuário escreve uma sequência de linhas de comando usando um editor de texto qualquer (fora do GrADS) e depois salva esse programa, por exemplo, com o nome de *programa1.gs*. O arquivo *programa1.gs* é definido como um script (a extensão **.gs** seria a sigla para **g**rads **s**cript) a ser executado dentro do prompt do GrADS.

O comando para executar um script, dentro do prompt do GrADS, é:

ga> run nome-do-script.gs

ou

ga> nome-do-script

Observações:

- Cada linha do script deve estar contida entre ' (apóstrofos), conforme exemplo abaixo:

```
* script feito pelo fulano; serve para mostrar o campo de temperatura
' open exemplo.ctl '
' d temp '
```

- Dentro dos scripts, as linhas iniciadas com o símbolo * são interpretadas como comentários (ver exemplo acima)

- O usuário pode também escrever um script sem utilizar os apóstrofos, porém a execução do mesmo é feita através do comando: **ga> exec nome-do-script.gs**

* Execução automática de scripts: **set imprun**

O comando **ga> set imprun nome-do-script.gs** executa automaticamente o mesmo antes de um comando **ga> d variável** conforme exemplo abaixo

Exemplo 38 – Começando a criar uma biblioteca de scripts, para facilitar/agilizar nossa vida no prompt do GrADS...

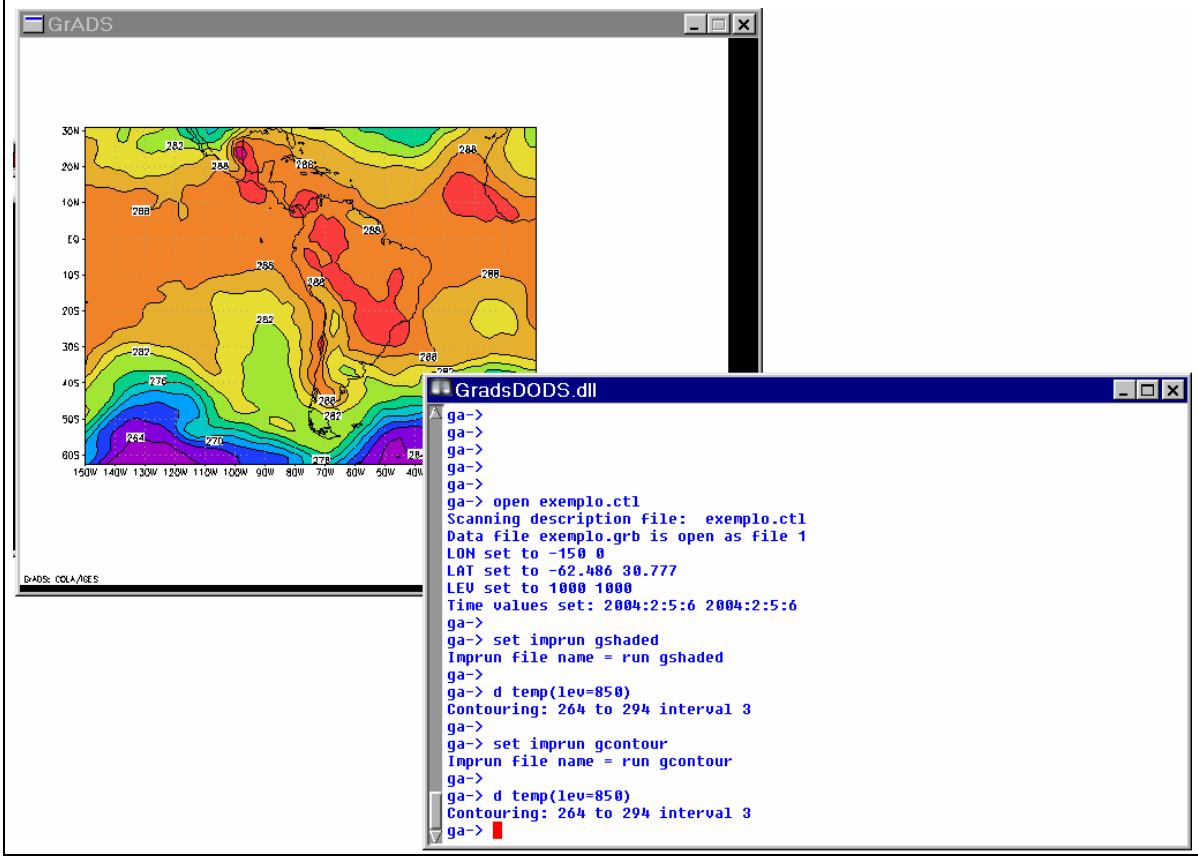
Abra seu editor de texto e escreva/salve o gshaded.gs (comandos abaixo)

```
* script feito pelo Everaldo: setagem para shaded de temperatura
'set gxout shaded'
'set ccols  auto'
```

Abra seu editor de texto e escreva/salve o gcontour.gs (comandos abaixo)

```
* script feito pelo Everaldo: setagem para contour de temperatura
' set  gxout contour '
' set ccolor 1 '
' set clab on '
' set clskip 2 '
```

Ok… agora carregue o GrADS em portrait e execute os comandos conforme a figura abaixo mostra... veja que a seqüência de comandos ficou mais "limpa"...



Exemplo 39 – Usando novo CTL (Dados de precipitação e role observados no mês de março entre os anos de 1975 a 1999 = 25 anos)

Arquivo olr-march.ctl:

```
dset \march\olr-march.bin
title OLR
undef -9999
xdef   145 linear   30.00  2.5
ydef    73 linear  -90.00  2.5
zdef 1 linear  1 1
tdef 25 linear 01mar1975 1yr
vars 1
olr   0 0 olr
ENDVARS
```

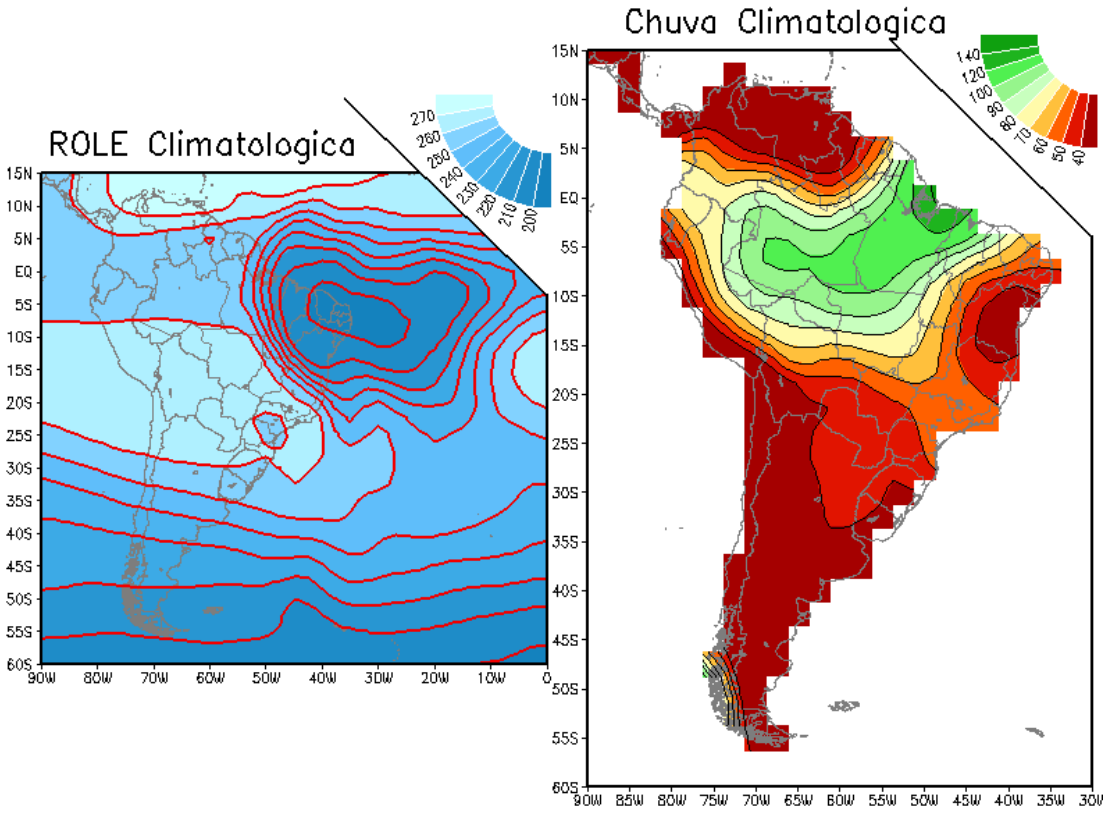
Arquivo gauge-march.ctl:

```
DSET \march\gauge-march.bin
UNDEF  -999.0
TITLE  Monthly Rainfall of Gauges
XDEF 144 LINEAR 1.25 2.5
YDEF 72  LINEAR -88.75 2.5
ZDEF  1 LEVELS  1
TDEF 25 LINEAR 01mar1975 1yr
VARS 1
rain 1 00 monthly rainfall (0.1 mm/day)
ENDVARS
```

Escrever um script (exemplo39.gs) com definição de novas cores, calculando média climatológica, rodando scripts (cbarc.gs, cores.gs) dentro do ex39.gs, colocando comentários, etc

```
'reinit'
'open \march\gauge-march.ctl'

* script de novas cores
'\march\cores'

* regiao da Am. do Sul
'set lat -60 15'; 'set lon -90 -30'

*definindo climatologia nos 25 anos
'define clichu=ave(rain.1, t=1, t=25)'

* plot da chuva
'set parea  5.9  10.9   0   8.5 '
'set grads off'; 'set grid off'
'set mpdset hires brmap'; 'set map 15 1 1'

'set gxout shaded'
'set ccols   29  27  25  23  21  32  34  36  38  39'
'set clevs   40  50  60  70  80  90 100 120 140'
'd smth9(clichu)'

'set gxout contour'; 'set clab off'; 'set ccolor 1'
'set clevs   40  50  60  70  80  90 100 120 140'
'd smth9(clichu)'

'draw title Chuva Climatologica      '
'cbarc 10.9 8.1'
'set parea off'

* fechar CTL 1
'close 1'

'open \march\olr-march.ctl'

* regiao da Am. do Sul
'set lat -60 15'; 'set lon 270 360'

*definindo climatologia nos 25 anos
'define cliolr=ave(olr.1, t=1, t=25)'

* plot da chuva
'set parea  0.5   5.5   0    8.5'
'set gxout shaded'
'set ccols   49  48  47  46  45  44  43  42  41  '
'set clevs  200 210 220 230 240 250 260 270'
'd smth9(cliolr)'

'set gxout contour'; 'set clab off'; 'set cthick 6'; 'set ccolor 2'
'set clevs  200 210 220 230 240 250 260 270'
```

```
'd smth9(cliolr)'

'\march\cbarc 5.5 7.5'

'draw title ROLE Climatologica          '
'set parea off'

*gerando arquivo de saida GIF
'printim \march\ex39.gif gif white'
```



6.2. Elementos de Linguagem nos Scripts

Em geral, os scripts contem os seguintes elementos:

comment
statement
assignment
say / prompt / pull
if / else / endif
while / endwhile / break / continue
function header / return

* Comment comentários dentro dos scripts devem conter o * na primeira coluna.

* Statement são as declarações de linhas de comando (expressões em geral)

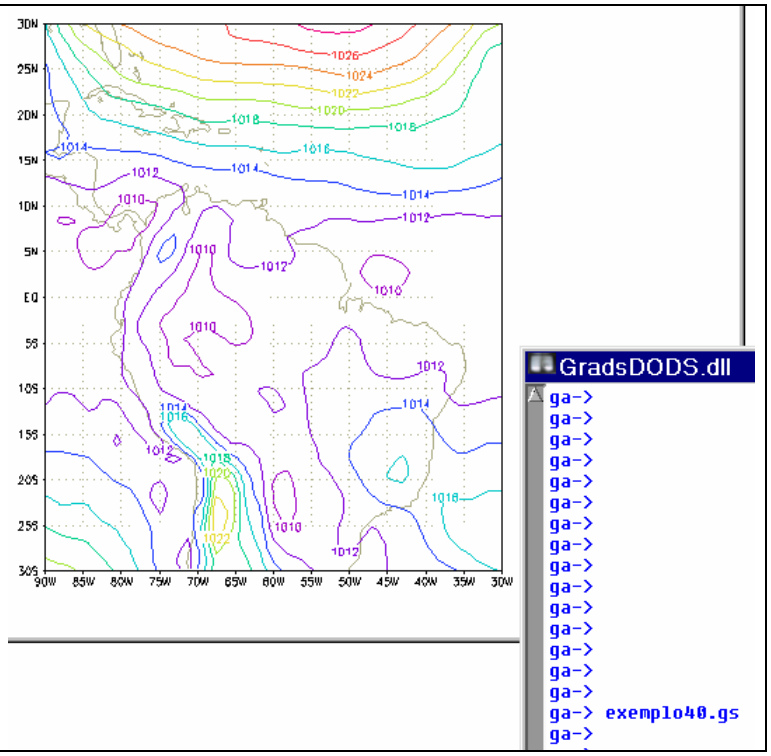
* Assignment é uma alegação usada em geral para definir uma variável = expressão

* Concatenação

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

Exemplo 40: Escreva/Salve as seguintes linhas de comando num exemplo40.gs e depois execute-o no GrADS... o resultado está na figura ao lado.

```
'open exemplo.ctl'
minlat = -30
maxlat = minlat + 60
minlon = -90
maxlon = -30
'set lat 'minlat%' '%maxlat
'set lon 'minlon' 'maxlon
'd psnm'
```

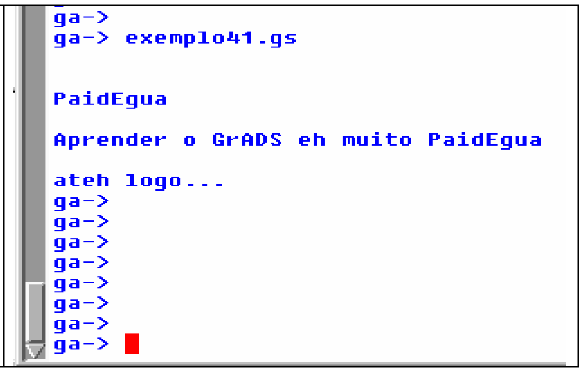


* say / prompt usados para fornecer informações ou fazer questionamentos ao usuário via terminal (prompt do GrADS), conforme sintaxe abaixo:

say 'expressão'
prompt expressão

Exemplo 41: Escreva/Salve as seguintes linhas de commando num exemplo41.gs e depois execute-o no GrADS... o resultado está na figura ao lado.

```
expressao = 'PaidEgua'
say line
say ' '
say 'Aprender o GrADS eh muito 'expressao
say ' '
say 'ateh logo... '
```

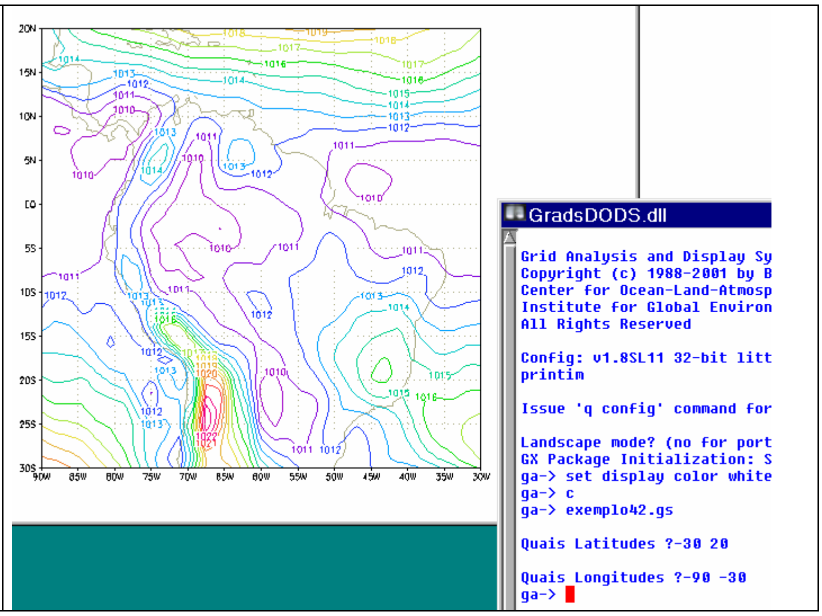


* pull carrega a informação fornecida pelo usuário via teclado, com a sintaxe:

pull variável

Exemplo 42: Escreva/Salve as seguintes linhas de comando no exemplo42.gs e depois execute-o no GrADS... o resultado está na figura ao lado.

```
'open exemplo.ctl'
prompt 'Quais Latitudes ?'
pull minlat maxlat
prompt 'Quais Longitudes ?'
pull minlon maxlon
'set lat 'minlat%' '%maxlat
'set lon 'minlon' 'maxlon
'd psnm'
```



* if / else / endif uma forma de controlar a execução do script... a sintaxe é:

```
if expressão
    linha de comando
    .
else
    linha de comando
    .
endif
```

Exemplo:

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

* while / endwhile uma forma de controlar a execução do script... a sintaxe é:

```
while
    linha de comando
    .
enwhile
```

Exemplo 43: Fazendo um loop no tempo…

```
'open \march\gauge-march.ctl'
tt = 1
while (tt <= 25)
  'set t 'tt
  'd rain'
  'c'
  tt = tt + 1
endwhile
```

* Variável global são variáveis que são mantidas ao longo de todo o script... sintaxe:

```
_var1 = 'variable-global1'
```

* Operadores:

```
|     logical OR
&     logical AND
!     unary NOT
-     unary minus
=     equal
!=    not equal
```

> greater than
>= greater than or equal
< less than
<= less than or equal
% concatenation
+ addition
- subtraction
*** multiplication**
/ division

* Funções intrínsecas

sublin (string, n) Armazena linha n extraída de um string de várias linhas

subwrd (string, n) Armazena uma palavra n extraída de um string

substr (string, start, length) Armazena uma parte de um string

* Comandos complementares

query <opções>
ou
q <opções>

As opções são:

q define lista todas variáveis definidas

q defval ival jval Fornece o valor do ponto de grade em ival, jval

q dims Fornece a dimensão do ambiente corrente

q file n Fornece informações do n arquivo CTL aberto

q files Lista CTL abertos

q fwrite Fornece o nome do arquivo usado na operação de fwrite

q gxinfo Lista as setagens gráficas

q pos Espera o click do maouse na tela de visualização, retornando a posição X,Y da tela

q shades Fornece níveis e cores setados na opção shaded

q time Fornece as informações da dimensão tempo

q transform coord1 coord2 transformações de coordenadas, onde o transform pode ser:

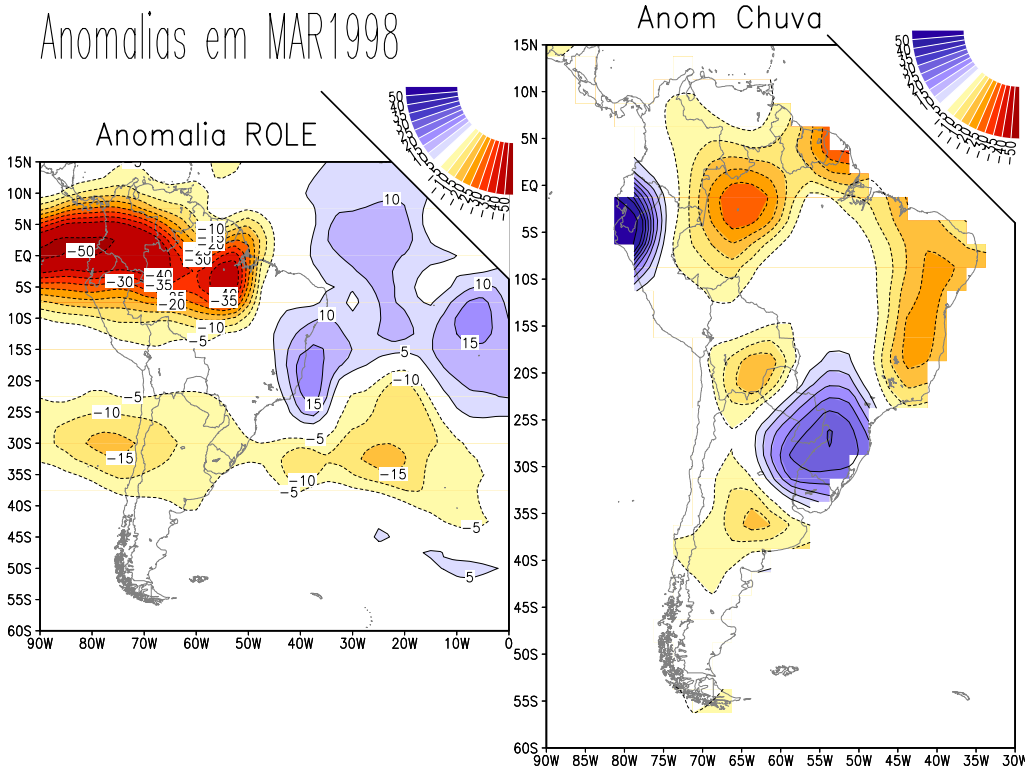
xy2w	XY coords to world coords
xy2gr	XY coords to grid coords
w2xy	world coords to XY coords
w2gr	world coords to grid coords
gr2w	grid coords to world coords
gr2xy	grid coords to XY coords

6.3. Exemplos e Exercícios

Exemplo 44: Calculando climatologia e plotando anomalias…

```
'reinit'
'open \march\gauge-march.ctl'

* script de novas cores
'\march\cores'

* regiao da Am. do Sul
'set lat -60 15'; 'set lon -90 -30'

*definindo climatologia nos 25 anos
'define clichu=ave(rain.1, t=1, t=25)'

* plot da anomalia de chuva
'set parea  5.9  10.9   0   8.5 '
'set grads off'; 'set grid off'
'set mpdset hires brmap'; 'set map 15 1 1'
'set t 24'
'set gxout shaded'
'set ccols   29  28  27  26  25  24  23  22 21 0 51 52 53 54 55 56 57 58 59'
'set clevs  -50 -40 -35 -30 -25 -20 -15 -10 -5 5 10 15 20 25 30 35 40 50'
'd smth9(rain.1(time=mar1998)-clichu)'
'set gxout contour'; 'set clab off'; 'set ccolor 1'
'set clevs  -50 -40 -35 -30 -25 -20 -15 -10 -5 5 10 15 20 25 30 35 40 50'
'd smth9(rain.1(time=mar1998)-clichu)'
'draw title Anom Chuva      '

'\march\cbarc 10.9 8.1'
'set parea off'

* fechar CTL 1
'close 1'
*---------------------------------------------------------------
'open \march\olr-march.ctl'
* regiao da Am. do Sul
```

```
'set lat -60 15'; 'set lon 270 360'

*definindo climatologia nos 25 anos
'define cliolr=ave(olr.1, t=1, t=25)'

'set t 24'
* plot da chuva
'set parea  0.5   5.5   0    8.5'
'set gxout shaded'
'set ccols   29  28  27  26  25  24  23  22 21 0 51 52 53 54 55 56 57 58 59'
'set clevs  -50 -40 -35 -30 -25 -20 -15 -10 -5 5 10 15 20 25 30 35 40 50'
'd smth9(olr.1(time=mar1998)-cliolr)'
'set gxout contour'; 'set clab on'; 'set ccolor 1'
'set clevs  -50 -40 -35 -30 -25 -20 -15 -10 -5 5 10 15 20 25 30 35 40 50'
'd smth9(olr.1(time=mar1998)-cliolr)'

'\march\cbarc 5.5 7.5'
'draw title Anomalia ROLE       '
'set parea off'

'q time'
res = subwrd(result,3)
mesano = substr(res,6,7)
'set strsiz 0.2 0.5'
'draw string 0.5 8.1 Anomalias em 'mesano

*gerando arquivo de saida GIF
'printim \march\ex44.gif gif white'
```



7. TÓPICOS ADICIONAIS

7.1. A Opção Template

7.2. Gerando Arquivos Binários com o FWRITE

7.3. Criando uma Máscara

7.4. UDF's

APÊNDICE A1: DESCRIÇÃO COMPLETA DE CADA COMPONENTE DO ARQUIVO DESCRITOR

DSET
DTYPE
INDEX
STNMAP
TITLE
UNDEF
UNPACK
FILEHEADER
THEADER
XYHEADER
XVAR
YVAR
ZVAR
STID
TVAR
TOFFVAR
OPTIONS
PDEF
XDEF
YDEF
ZDEF
TDEF
VARS
ENDVARS

DSET *data_filename*

This entry specifies the filename of the data file being described. If the data and the descriptor file are not in the same directory, then *data_filename* must include a full path. If a ^ character is placed in front of *data_filename*, then *data_filename* is assumed to be relative to the path of the descriptor file. If *data_filename* does not include a full path or a ^, then GrADS assumes the data set and its descriptor file are in the same directory. If you are using the ^ character in the DSET entry, then the descriptor file and the data file may be moved to a new directory without changing any entries in the data descriptor file, provided their relative paths remain the same. For example:
If the data descriptor file is: /data/wx/grads/sa.ctl and the binary data file is: /data/wx/grads/sa.dat
then the data file name in the data descriptor file can be:

```
DSET ^sa.dat
```

instead of:

```
DSET /data/wx/grads/sa.dat
```

DTYPE *keyword*

The DTYPE entry specifies the type of data being described. There are four options: grib, hdfsds, netcdf, or station. If the data type is none of these, then the DTYPE entry is omitted completely from the descriptor file and GrADS will assume the data type is gridded binary.

bufr	(GrADS version 1.9) Data file is a BUFR station data file. This data type must be accompanied by the following special entries: XVAR, YVAR, TVAR, STID. Optional special entries are: ZVAR, TOFFVAR.
grib	Data file is an indexed GRIB file. This data type requires a secondary entry in the descriptor file: INDEX. The INDEX entry provides the filename (including the full path or a ^) for the grib index file. The index file is created by the gribmap utility. You must run gribmap and create the index file before you can display the grib data in GrADS.
hdfsds	(GrADS version 1.9) Data file is an HDF Scientific Data Set (SDS). Although HDF-SDS files are self-describing and may be read automatically using the sdfopen/xdfopen commands, this DTYPE gives you the option of overriding the file's own metadata and creating a descriptor file for some or all of the variables in the file. This DTYPE may also be used if the metadata in the HDF-SDS file is insufficient or is not coards-compliant. This data type requires a special entry in the *units* field of the variable declaration. The undef and unpack entries contain special options for this dtype.
netcdf	(GrADS version 1.9) Data file is NetCDF. Although NetCDF files are self-describing and may be read automatically using the sdfopen/xdfopen commands, this DTYPE gives you the option of overriding the file's own metadata and creating a descriptor file for some or all of the variables in the file. This DTYPE may also be used if the metadata in the NetCDF file is insufficient or is not coards-compliant. This data type requires a special entry in the *units* field of the variable declaration. The undef and unpack entries contain special options for this dtype.
station	Data file is in GrADS station data format. This data type requires a secondary entry in the descriptor file: STNMAP. The STNMAP entry provides the filename (including the full path or a ^) for the station data map file. The map file is created by the stnmap utility. You must run stnmap and create the map file before you can display the station data in GrADS.

INDEX *filename*

This entry specifies the name of the grib map file. It is required when using the DTYPE grib entry to read grib formatted data. The file is generated by running the external utility gribmap. Filenaming conventions are the same as those described for the DSET entry.

STNMAP *filename*

This entry specifies the name of the station map file. It is required when using the DTYPE station entry to read GrADS-formatted station data. The file is generated by running the external utility stnmap. Filenaming conventions are the same as those described for the DSET entry.

TITLE *string*

This entry gives brief description of the contents of the data set. *String* will be included in the output from a query command and it will appear in the directory listing if you are serving this data file with the GrADS-DODS Server (GDS), so it is helpful to put meaningful information in the title field. For GDS use, do not use double quotation marks (") in the title.

UNDEF *value <undef_attribute_name>*

This entry specifies the undefined or missing data value. UNDEF is a *required entry* even if there are no undefined data. GrADS operations and graphics routines will ignore data with this value from this data set.
(GrADS version 1.9) An optional second argument has been added for data sets of DTYPE netcdf or hdfsds -- it is the name of the attribute that contains the undefined value. This is used when individual variables in the data file have different missing values. Attribute names are case sensitive, and it is assumed that the name is identical for all variables in the NetCDF or HDF-SDS data file. After data I/O, the missing values in the grid are converted from the individual undef to the file-wide undef (the numerical value in the first argument of the undef record). Then it appears to GrADS that all variables have the same undef, even if they don't in the original data file. If the name given does not match any attributes, or if no name is given, the file-wide undef value will be used.
Example: UNDEF 1e+33 _FillValue

UNPACK *scale_factor_attribute_name add_offset_attribute_name*

(GrADS version 1.9) This entry is used with DTYPE netcdf or hdfsds for data variables that are 'packed' -- i.e. non-float data that need to be converted to float by applying the following formula:

y = x * *scale_factor* + *add_offset*

Two arguments are required, the first is the attribute name for the scale factor (e.g. scale_factor, Slope), the second is the attribute nme for the offset (e.g. add_offset, Intercept). Attribute names are case sensitive, and it is assumed that the names are identical for all variables in the NetCDF or HDF-SDS data file. If the names given do not match any attributes, the scale factor will be assigned a value of 1.0 and the offset will be assigned a value of 0.0.
Example: UNPACK scale_factor add_offset

FILEHEADER *length*

This optional entry tells GrADS that your data file has a header record of *length* bytes that precedes the data. GrADS will skip past this header, then treat the remaineder of the file as though it were a normal GrADS binary file after that point. This optional descriptor file entry is only valid for GrADS gridded data sets.

THEADER *length*

This optional entry tells GrADS that the data file has a header record of *length* bytes preceding each time block of binary data. This optional descriptor file entry is only valid for GrADS gridded data sets. See the section on structure of a gridded binary data file for more information.

XYHEADER *length*

This optional entry tells GrADS that the data file has a header record of length bytes preceding each horizontal grid (XY block) of binary data. This optional descriptor file entry is only valid for GrADS gridded data sets. See the section on structure of a gridded binary data file for more information.

XVAR *x,y*

(GrADS version 1.9) This entry provides the x,y pair for the station's longitude. This entry is required for DTYPE bufr.

YVAR *x,y*

(GrADS version 1.9) This entry provides the x,y pair for the station's latitude. This entry is required for DTYPE bufr.

ZVAR *x,y*

(GrADS version 1.9) This entry provides the x,y pair for the station data's vertical coordinate (e.g., pressure). This is an optional entry for DTYPE bufr.

STID *x,y*

(GrADS version 1.9) This entry provides the x,y pair for the station ID. This entry is required for DTYPE bufr.

TVAR yr *x,y* mo *x,y* dy *x,y* hr *x,y* mn *x,y* sc *x,y*

(GrADS version 1.9) This entry provides the x,y pairs for all the **base time** coordinate variables. Each time unit (year=yr, month=mo, day=dy, hour=hr, minute=mn, second=sc) is presented as a 2-letter abbreviation followed by the x,y pair that goes with that time unit. The time for any individual station report is the base time plus the offset time (see TOFFVAR). All six base time units are not required to appear in the TVAR record, only those that are in the data file. This entry is required for DTYPE bufr.

TOFFVAR yr *x,y* mo *x,y* dy *x,y* hr *x,y* mn *x,y* sc *x,y*

(GrADS version 1.9) This entry provides the x,y pairs for all the **offset time** coordinate variables. The syntax is the same as TVAR. The time for any individual station report is the base time plus the offset time. All six offset time units are not required to appear in the TOFFVAR record, only those that are in the data file. This is an optional entry for DTYPE bufr.

OPTIONS *keyword*

This entry controls various aspects of the way GrADS interprets the raw data file. It replaces the old FORMAT record. The *keyword* argument may be one or more of the following:

yrev	Indicates that the Y dimension (latitude) in the data file has been written in the reverse order from what GrADS

	assumes. An important thing to remember is that GrADS still presents the view that the data goes from south to north. The YDEF statement does not change; it still describes the transformation from a grid space going from south to north. The reversal of the Y axis is done as the data is read from the data file.
zrev	Indicates that the Z dimension (pressure) in the data file has been written from top to bottom, rather than from bottom to top as GrADS assumes. The same considerations as noted above for yrev also apply.
template	Indicates that a template for multiple data files is in use. For more information, see the section on Using Templates.
sequential	Indicates that the file was written in sequential unformatted I/O. This keyword may be used with either station or gridded data. If your gridded data is written in sequential format, then each record must be an X-Y varying grid. If you have only one X and one Y dimension in your file, then each record in the file will be one element long (it may not be a good idea to write the file this way).
365_day_calendar	Indicates the data file was created with perpetual 365-day years, with no leap years. This is used for some types of model ouput.
byteswapped	Indicates the binary data file is in reverse byte order from the normal byte order of your machine. Putting this keyword in the OPTIONS record of the descriptor file tells GrADS to swap the byte order as the data is being read. May be used with gridded or station data.

The best way to ensure hardware independence for gridded data is to specify the data's source platform. This facilitates moving data files and their descriptor files between machines; the data may be used on any type of hardware without having to worry about byte ordering. The following three OPTIONS keywords are used to describe the byte ordering of a gridded or station data file:

big_endian	Indicates the data file contains 32-bit IEEE floats created on a big endian platform (e.g., sun, sgi)
little_endian	Indicates the data file contains 32-bit IEEE floats created on a little endian platform (e.g., iX86, and dec)
cray_32bit_ieee	Indicates the data file contains 32-bit IEEE floats created on a cray.

PDEF

PDEF is so powerful it has its own documentation page.

XDEF *xnum mapping <additional arguments>*

This entry defines the grid point values for the X dimension, or longitude. The first argument, *xnum*, specifies the number of grid points in the X direction. *xnum* must be an integer >= 1. *mapping* defines the method by which longitudes are assigned to X grid points. There are two options for *mapping*:
LINEAR Linear mapping
LEVELS Longitudes specified individually
The LINEAR mapping method requires two additional arguments: *start* and *increment*. *start* is a floating point value that indicates the longitude at grid point X=1. Negative values indicate western longitudes. *increment* is the spacing between grid point values, given as a positive floating point value.
The LEVELS mapping method requires one additional argument, *value-list*, which explicitly specifies the longitude value for each grid point. *value-list* should contain *xnum* floating point values. It may continue into the next record in the descriptor file, but note that records may not have more than 255 characters. There must be at least 2 levels in *value-list*; otherwise use the LINEAR method.
Here are some examples:

```
XDEF   144   LINEAR   0.0 2.5
XDEF    72   LINEAR   0.0 5.0
XDEF    12   LEVELS   0 30 60 90 120 150 180 210 240 270 300 330
XDEF    12   LEVELS   15 45 75 105 135 165 195 225 255 285 315 345
```

YDEF *ynum mapping <additional arguments>*

This entry defines the grid point values for the Y dimension, or latitude. The first argument, *ynum*, specifies the number of grid points in the Y direction. *ynum* must be an integer >= 1. *mapping* defines the method by which latitudes are assigned to Y grid points. There are several options for *mapping*:
LINEAR Linear mapping
LEVELS Latitudes specified individually
GAUST62 Gaussian T62 latitudes
GAUSR15 Gaussian R15 latitudes
GAUSR20 Gaussian R20 latitudes
GAUSR30 Gaussian R30 latitudes
GAUSR40 Gaussian R40 latitudes
The LINEAR mapping method requires two additional arguments: *start* and *increment*. *start* is a floating point value that indicates the latitude at grid point Y=1. Negative values indicate southern latitides. *increment* is the spacing between grid point values in the Y direction. It is assumed that the Y dimension values go from south to north, so *increment* is always positive.
The LEVELS mapping method requires one additional argument, *value-list*, which explicitly specifies the latitude for each grid point, from south to north. *value-list* should contain *ynum* floating point values. It may continue into the next record in the descriptor file, but note that records may not have more than 255 characters. There must be at least 2 levels in *value-list*; otherwise use the LINEAR method.
The Gaussian mapping methods require one additional argument: start. This argument indicates the first gaussian grid number. If the data span all latitudes, start would be 1, indicating the southernmost gaussian grid latitude.
Here are some examples:

```
YDEF    73   LINEAR   -90 2.5
YDEF   180   LINEAR   -90 1.0
YDEF    18   LEVELS   -85 -75 -65 -55 -45 -35 -25 -15 -5 5 15 25 35 45 55 65 75 85
```

```
YDEF     94   GAUST62   1
YDEF     20   GAUSR40   15
```

The NCEP/NCAR Reanalysis surface variables are on the GAUST62 grid.
The final example shows that there are 20 Y dimension values which start at Gaussian Latitude 15 (64.10 south) on the Gaussian R40 grid

ZDEF *znum mapping <additional arguments>*

This entry defines the grid point values for the Z dimension. The first argument, *znum*, specifies the number of pressure levels. *znum* must be an integer >= 1. *mapping* defines the method by which longitudes are assigned to Z grid points. There are two options for *mapping*:
LINEAR Linear mapping
LEVELS Pressure levels specified individually
The LINEAR mapping method requires two additional arguments: *start* and *increment*. *start* is a floating point value that indicates the longitude at grid point Z=1. *increment* is the spacing between grid point values in the Z direction, or from lower to higher. *increment* may be a negative value.
The LEVELS mapping method requires one additional argument, *value-list*, which explicitly specifies the pressure level for each grid point in ascending order. *value-list* should contain *znum* floating point values. It may continue into the next record in the descriptor file, but note that records may not have more than 255 characters. There must be at least 2 levels in *value-list*; otherwise use the LINEAR method.
Here are some examples:

```
ZDEF    10  LINEAR   1000 -100
ZDEF     7  LEVELS   1000 850 700 500 300 200 100
ZDEF    17  LEVELS   1000 925 850 700 600 500 400 300 250 200 150 100 70 50
```

TDEF *tnum* LINEAR *start increment*

This entry defines the grid point values for the T dimension. The first argument, *tnum*, specifies the number of time steps. *tnum* must be an integer >= 1. The method by which times are assigned to T grid points is always LINEAR.
start indicates the initial time value at grid point T=1. *start* must be specified in the GrADS absolute date/time format:
hh:*mm*Z*dd*mmm*yyyy*
where:

```
hh      = hour (two digit integer)
mm      = minute (two digit integer)
dd      = day (one or two digit integer)
mmm     = 3-character month
yyyy    = year (may be a two or four digit integer; 2 digits implies a year between 1950 and 2049)
```

If not specified, *hh* defaults to 00, *mm* defaults to 00, and *dd* defaults to 1. The month and year must be specified. No intervening blanks are allowed in the GrADS absolute date/time format.
increment is the spacing between grid point values in the T direction. *increment* must be specified in the GrADS absolute time increment format:
vvkk
where:

```
vv    = an integer number, 1 or 2 digits
kk    = mn (minute)
        hr (hour)
        dy (day)
        mo (month)
        yr (year)
```

Here are some examples:

```
TDEF     60   LINEAR   00Z31dec1999 1mn
TDEF     73   LINEAR   3jan1989 5dy
TDEF    730   LINEAR   00z1jan1990 12hr
TDEF     12   LINEAR   1jan2000 1mo
TDEF    365   LINEAR   12Z1jan1959 1dy
TDEF     40   LINEAR   1jan1950 1yr
```

VARS *varnum*
variable_record_1
variable_record_2
...
variable_record_*varnum*
ENDVARS

This ensemble of entries describes all the variables contained in the data set. *varnum* indicates the number of variables in the data set and is therefore also equal to the number of variable records that are listed between the VARS and ENDVARS entries. ENDVARS must be the final line of the Grads data descriptor file. Any blank lines after the ENDVARS statement may cause open to fail!
The format of the variable records is as follows:
varname levs units description
The syntax of *varname* and *units* is different depending on what kind of data format (DTYPE) you are describing. Details provided below:

varname	This is a 1-15 character "name" or abbreviation for the data variable. *varname* may contain alphabetic and numeric characters but it must start with an alphabetic character (a-z).
varname (DTYPE netcdf or hdfsds)	(GrADS version 1.9) For DTYPE netcdf or hdfsds, *varname* may have a different syntax: SDF_varname=>grads_varname SDF_varname is the name the data variable was given when the SDF file was originally created. For NetCDF files, this name appears in the output from ncdump. It is important that SDF_varname exactly matches the variable name in the data file. SDF_varname may contain uppercase letters and non-alpha-numeric characters. The classic *varname* syntax (i.e., when "SDF_varname =>" is omitted) may be used if SDF_varname meets the criteria for GrADS variable names: it must be less than 15 characters, start with an alphabetic character, and cannot contain any upper case letters or non-alpha-numeric characters.

levs	This is an integer that specifies the number of vertical levels the variable contains. *levs* may not exceed *znum* as specified in the ZDEF statement. If *levs* is 0, the variable does not correspond to any vertical level. Surface variables (e.g. sea level pressure) have a *levs* value of 0. For DTYPE station or bufr, surface variables have a *levs* value of 0 and upper air variables have a *levs* value of 1. (Exception to this rule for bufr data: replicated surface variables are given a levs value of 2).
description	This is text description or long name for the variable, max 40 characters.

The *units* component of the variable record is used for data with DTYPE bufr, grib, netcdf, or hdfsds. It is also used for non-standard binary data files that require special "unpacking" instructions, and special cases of pre-projected wind components. If the data you are describing does not fall into any of these categories, put a value of 99 in the *units* field.

units	For flat binary files containing 4-byte floating-point data that are not pre-projected, this field is ignored but must be included. Put in a value of 99.
units (DTYPE bufr)	(GrADS version 1.9) For DTYPE bufr files, this field contains the x,y pair for the named variable.
units (DTYPE grib)	For DTYPE grib, the *units* field specifies the GRIB parameters of the variable. This information is used by the gribmap utility for mapping the variables listed in the descriptor file to the data records in the GRIB files. This parameter may contain up to four comma-delimited numbers: VV,LTYPE,LEVEL,RI where, VV = The GRIB parameter number (Required) LTYPE = The level type indicator (Required) LEVEL = The value of the LTYPE (Optional) RI = The "range indicator" (for certain level types) (Optional) The external utilities gribscan and wgrib are quite useful in determining what the values for the *units* field should be for a GRIB data file. Examples: u 39 33,100 U Winds [m/s] t 39 11,100 Temperature [K] ts 0 11,1 Surface Temperature [K] tb 0 11,116,60,30 Temperature, 30-60mb above surface [K] dpt 0 17,100,1000 Dew Point Temperature at 1000 mb [K]
units (DTYPE netcdf or hdfsds)	(GrADS version 1.9) For DTYPE netcdf or hdfsds, the *units* field is a comma-delimited list of the varying dimensions of the variable. Dimensions expressed as x, y, z, or t correspond to the four axes defined by XDEF, YDEF, ZDEF and TDEF. For example, a surface variable such as sea level pressure might look like this: presSFC=>psfc 0 y,x Surface Pressure A time-varying atmospheric variable such as geopotential height might look like this: Height=>hght 17 t,z,y,x Geopotential Height (m) The order of the dimensions listed in the *units* field does matter. They must describe the shape of the variable as it was written to the SDF data file. For NetCDf files, this information appears in the output from ncdump next to the variable name. If your data file contains a variable that also varies in a non-world-coordinate dimension (e.g. histogram interval, spectral band, ensemble number) then you can put a non-negative integer in the list of varying dimensions that will become the array index of the extra dimension. For example: VAR=>hist0 0 0,y,x First historgram interval for VAR VAR=>hist1 0 1,y,x Second historgram interval for VAR VAR=>hist2 0 2,y,x Third histogram interval for VAR Another option in this example would be to fill the unused Z axis with the histogram intervals: zdef 3 linear 1 1 ... VAR=>hist 0 z,y,x VAR Histogram In this case, it would appear to GrADS that variable 'hist' varies in Z, but the user would have to remember that the Z levels correspond to histogram intervals. The latter technique makes it easier to slice through the data, but is not the most accurate representation. And if you don't have an unsued world-coordinate axis available, then you still have a way to access your data.
units (non-standard binary)	For non-standard binary files, the *units* field is used to instruct GrADS how to read binary files that do not conform to the default structure or do not contain 4-byte float data. GrADS assumes the data were written in the following order (starting from the fastest varying dimension to the slowest): longitude (X), latitude (Y), vertical level (Z), variable (VAR), time (T). If your binary data set was created or "packed" according to a different dimension sequence, then you can use the *units* field to tell GrADS exactly how to unpack the data. For these non-standard binary files, the *units* field is a series of one or more comma-delimited numbers, the first of which is always -1. The syntax is as follows: -1, *structure* <,arg> There are four options for *structure*, outlined below. Some of these options have additional attributes which are specified with *arg*.

-1,10,*arg*	This option indicates that "VAR" and "Z" have been transposed in the dimension sequence. The order is: longitude (X), latitude (Y), variable (VAR), vertical level (Z), time(T). Thus, all variables are written out one level at a time. This feature was designed to be used with NASA GCM data in the "phoenix" format. The upper air *prognostic* variables were transposed, but the *diagnostic* variables were not. Thus an *arg* of 1 means the variable has been var-z transposed, and an *arg* of 2 means the variable has not.

	-1,20,*arg*	This option indicates that "VAR" and "T" have been transposed in the dimension sequence. The order is: longitude (X), latitude (Y), vertical level (Z), time(T), variable (VAR). Thus, all times for one variable are written out in order followed by all times for the next variable, etc. If your data set is actually a collection of separate files that are aggregated by using a template, then you must use *arg* to tell GrADS how many time steps are contained in each individual file. For example, here are the relevant records from a descriptor file for 10 years of monthly wind and temperature data packaged in 10 separate files (one for each year) with "VAR" and "T" dimensions transposed: DSET ^monthlydata_%y4.dat OPTIONS template TDEF 120 linear jan79 1mo VARS 3 u 17 -1,20,12 U-Wind Component v 17 -1,20,12 V-Wind Component t 17 -1,20,12 Temperature ENDVARS
	-1,30	This option handles the cruel and unusual case where X and Y dimensions are transposed and the horizontal grids are (lat,lon) as opposed to (lon,lat) data. This option causes GrADS to work very inefficiently. However, it is useful for initial inspection and debugging.
	-1,40,*arg*	This option handles non-float data. Data are converted to floats internally after they are read from the binary file. The dimension sequence is assumed to be the default. The secondary *arg* tells GrADS what type of data values are in the binary file: *units* = -1,40,1 = 1-byte unsigned chars (0-255) *units* = -1,40,2 = 2-byte unsigned integers *units* = -1,40,-2 = 2-byte signed integers *units* = -1,40,4 = 4-byte integers
units (pre-projected wind components)	For pre-projected vector component data that require the use of PDEF and rotation, GrADS has to retrieve both the u and v component in order to do the rotation calculation. GrADS determines how to match u and v variables by checking the *units* field of the variable record. The u variable must have a *units* value of 33, and the v variable must have a *units* value of 34. (This is the GRIB convention). If there are more than one u/v pairs, secondary *units* values are used.	