



L. Přech

Programování v IDL

zpracování a vizualizace dat

UK-MFF

Stručné texty ke kursu EVF088x1a

L. Přech 1/1 KZ

staženo z WEBu dne 13.12.2003 a zpracováno k tisku

- RIEMANN Computes the "Riemann sum" (or its inverse) which helps implement the backprojection operator used to reconstruct the cross-section of an object, given projections through the object from multiple directions.
- POLYSHADE Returns a shaded-surface representation (2D array) of one or more solids described by a set of polygons
- SHADE_VOLUME Computes the polygons that describe a three dimensional contour surface
- SEARCH3D Finds "objects" or regions of similar data values within a 3D array of data

Polygonální a tetrahedrální sítě.

- MESH_CLIP, MESH_DECIMATE, MESH_ISSOLID, MESH_MERGE, MESH_NUMTRIANGLES, MESH_OBJ, MESH_SMOOTH, MESH_SURFACEAREA, MESH_VALIDATE, MESH_VOLUME
- COMPUTE_MESH_NORMALS Computes normal vectors for a set of polygons described by the input array
- POLYSHADE Returns a shaded-surface representation (2D array) of one or more solids described by a set of polygons.
- ISOCONTOUR, ISOSURFACE
- TETRA_CLIP, TETRA_SURFACE, TETRA_VOLUME
- VERT_T3D Transforms a 3D array by a 4x4 transformation matrix and returns the transformed array

- HLS, HSV, PSEUDO - definice barevné škály spirálou v HLS, HSV resp. LHB (Lightness Hue Brightness) barevném systému
- COLOR_QUAN - převod true-color obrazu (24bit/pixel) na pseudo-color obraz (8bit/pixel) a odpovídající barevnou škálu (2-256 barev)
- REDUCE_COLORS - sníží počet barev v obraze eliminací barev nepoužitých pro pixely
- MULTI - upraví současnou barevnou škálu několikanásobným cyklickým přetočením
- GAMMA_CT - gama korekce barevné škály
- STRETCH - stlačí barevnou škálu do daného rozsahu indexů

Promítání obrazu dle zeměpisné projekce.

- MAP_IMAGE, MAP_PATCH deformuje obraz podle nastavené zeměpisné projekce

Field Tracing

Trasování vektorových polí.

- PARTICLE_TRACE - hledá cestu nehmotné částic ve 2D a 3D vektorovém poli
- STREAM_LINE Output a polygonal ribbon which is tangent to a vector field along its length.
- VECTOR_FIELD

Analytical Geometry

Rutiny použitelné v analytické geometrii.

- PNT_LINE - vzdálenost bodu od přímky
- CIR_3PNT - střed a poloměr kružnice ze 3 zadaných bodů (v 2D)
- SPH_4PNT - střed a poloměr kulové plochy ze 4 zadaných bodů (v 3D)
- POLY_AREA - plocha mnohoúhelníku v 2D

3D transformace.

- T3D
- SCALE3
- SCALE3D
- SURF

Volume and 3D Data Processing

Zpracování objemových dat.

- PROJECT_VOL - projekce 3D objemových dat (poloprůhledný prostor) do 2D roviny
- VOXEL_PROJ - projekce 3D objemových dat (poloprůhledný prostor) do 2D roviny
- RECON3 Can reconstruct a three-dimensional data array from two or more images (or projections) of an object

Úvod

IDL je Interactive Data Language (viz. firemní stránky Research Systems Inc.¹)

Kurs probíhá v zimním semestru v *počítačové učebně Troja* (PUC) v pátek 13:10-14:40. Pro práci v učebně a dolní laboratoři je k dispozici zhruba *12 plovoucích licencí* IDL ve verzi 6.0. Další instalace existují na Katedře elektroniky a vakuové fyziky².

Součástí nejnovější verze 6.0 je volně šiřitelný runtime IDL Virtual Machine³, který umožňuje spouštět přeložené aplikace i na strojích bez licence.

Prezentace IDL⁴ - Interactive Data Language je velmi kvalitní software pro analýzu dat, vizualizaci a vývoj cross-platform aplikací (Windows, Unix, Mac, VMS, ...). V integrovaném vývojovém prostředí IDL zahrnuje nástroje pro všechny typy projektů — od "quick-look", interaktivní analýzy a zobrazení až po velké komerční programové projekty.

Kurz je vhodný pro studenty všech fyzikálních oborů, zejména astronomie, fyzikální elektroniky, geofyziky, kosmické fyziky či meteorologie (v těchto disciplínách je IDL častým profesionálním nástrojem). IDL se může stát Vaším nástrojem pro fyzikální praktikum, diplomovou i disertační práci.

Předpokládají se elementární znalosti z programování v libovolném programovacím jazyce. Základní znalosti použití numerických metod jsou výhodou.

Zápočet bude udělen za *vypracování programu pro zpracování a prezentaci dat* dle dohody⁵.

Osnova kurzu

- Práce ve vývojovém prostředí IDL.
- Základní programové konstrukce, deklarace proměnných, funkcí a procedur.
- Datové formáty.
 - Práce s řetězci.
- Práce se soubory.
 - Textové a binární soubory.
 - High-level rutiny pro čtení textových a binárních souborů.
- Grafická výstupní zařízení, okna.
- 2D a 3D grafika, práce s barvou, fonty, tisk.
 - Čarové a bodové grafy.
 - Interaktivní rutiny pro 2D a 3D grafiku.
- Matematické algoritmy v IDL - přehled algoritmů, např. interpolace dat, fitování křivek a ploch, filtrace, možnosti analýzy signálu a zpracování obrazu, statistika ad.

¹ <http://www.rsinc.com/idl/>

² http://www.troja.mff.cuni.cz/fs_troja/kevf/home.html.cs

³ <http://www.rsinc.com/idlvm/enduser.asp>

⁴ <http://aurora.troja.mff.cuni.cz/texty/IDLprez/>

⁵ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/default.htm#ukaz#ukaz

- Další možnosti IDL - animace, zobrazení objemu, užití map a zeměpisných projekcí atd.
- Vytváření aplikací s grafickým uživatelským rozhraním.
- Přenositelnost na jiné platformy, integrace s dalšími programovacími jazyky aj.

Další literatura o IDL v anglickém jazyce

- Manuály IDL 5.3 tištěné - k prezenčnímu studiu u mne.
- Manuály IDL 5.3 až 6.0 ve formátu PDF nebo Windows CHM – po dohodě možnost zapůjčení CD.
- *David W. Fanning, IDL Programming Techniques*, 2nd ed., 2000⁶ – k prezenčnímu studiu též u mne.

Odkazy na IDL na WWW

- IDL related sites⁷ from Research Systems Inc.

Doplňkové knihovny IDL rutin

- Coyote's library⁸ (D.W. Fanning)
- knihovna Skupiny kosmické fyziky KEVF

IDL – jak na to?

Zde se postupně objeví jednoduché příklady řešení typických problémů, které studentům dělávají potíže. Netvrdím, že stejná věc nepůjde udělat ještě jinými způsoby, případně efektivněji. Můžete posílat tipy, co by tu mělo být.

Procvičte si IDL

Tématická cvičeníčka.

Ukázky zápočtových programů

- Zpracování experimentálních vybíjecích křivek⁹ prachových nabitých částic (ZS 2000/2001, autor: *Jiří Pavlů*)
- Rekonstrukce topografie měsíčního povrchu¹⁰ z fotografií (ZS 2000/2001, autor: *Martin Švec*)
- Zpracování série snímků úplného zatmění Slunce¹¹ (ZS 2000/2001, autor: *Michal Švanda*)
- Využití IDL pro zobrazení výsledků numerických výpočtů¹² (ZS 2000/2001, autor: *Richard Wunsch*)

⁶ <http://www.dfanning.com>

⁷ <http://www.rsinc.com/related/index.cfm>

⁸ <http://www.dfanning.com/documents/programs.html>

⁹ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/pavlu/nabijeni.htm

¹⁰ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/svec/index.htm

¹¹ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/svanda/index.html

¹² http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/wunsch/default.htm

- ROBERTS Roberts gradient of an image.
- SOBEL Sobel gradient operator.
- LEEFILT Performs the Lee filter algorithm on an image array using a box of size $2N+1$.
- THIN Returns the "skeleton" of a bi-level image.
- DISSOLVE Provides a digital "dissolve" effect for images.
- CONVOL Discrete convolution of image with a selected kernel.
- HIST_EQUAL Histogram equalisation of image, returns a histogram-equalized byte array.
- H_EQ_CT Histogram-equalizes the color tables for an image or a region of the display.
- H_EQ_INT Interactively histogram-equalizes the color tables of an image or a region of the display
- ADAPT_HIST_EQUAL Performs adaptive histogram equalization, a form of automatic image contrast enhancement
- DILATATE, ERODE Morphologic dilation and erosion operators on binary and grayscale images and vectors
- MORPH_OPEN, MORPH_CLOSE Opening and closing operators to a binary or grayscale image
- MORPH_DISTANCE Estimates N-dimensional distance maps
- MORPH_GRADIENT Applies the morphological gradient operator to a grayscale image
- MORPH_HITORMISS, MORPH_THIN Applies the hit-or-miss operator, thinning operation to a binary image
- MORPH_TOPHAT Applies top-hat operator to a grayscale image
- POLY_2D Performs polynomial warping of images
- POLYWARP Determines coefficients of the polynomial functions performing polynomial spatial warping.
- WATERSHED Applies the morphological watershed operator to a grayscale image.
- WARP_TRI Image array with a specified geometric correction applied

Regions of interest.

- POLYFILLV Returns a vector containing the one-dimensional subscripts of the array elements contained inside a polygon defined by vectors X and Y
- POLY_AREA Area of a polygon given the coordinates of its vertices
- DEFROI Interactively defines an irregular region of interest of an image using the image display system and the cursor and mouse
- PROFILE Extracts a profile from an image and returns a floating-point vector containing the values of the image along the profile line marked by the user
- SEARCH2D Finds "objects" or regions of similar data values within a 2D array of data.
- SEARCH3D Finds "objects" or regions of similar data values within a 3D array of data.

Kódování barev, barevné škály.

- COLOR_CONVERT - převod mezi barevnými soustavami RGB (Red Green Blue), HLS (Hue Lightness Saturation) a HSV (Hue Saturation Value)

- READ_SPR Read a row-indexed sparse array from a file.
- SPRSAB Multiply two row-indexed sparse arrays.
- SPRSAX Multiply a row-indexed sparse array by a vector.
- SPRSIN Convert an array or list to row-indexed sparse storage format.
- WRITE_SPR Write a row-indexed sparse array to a file.

Routines for Time-Series Analysis

Zpracování časových řad.

- A_CORRELATE Compute the autocorrelation or autocovariance of a sample population.
- C_CORRELATE Compute the cross-correlation or cross-covariance of two sample populations.
- SMOOTH Smooth a time-series using a moving average.
- TS_COEF Compute the coefficients used in an autoregressive time-series forecasting model.
- TS_DIFF Compute forward differences of a time-series.
- TS_FCAST Compute the future values of a stationary time-series.
- TS_SMOOTH Compute central, backward, or forward moving averages of a time-series.

Routines for Signal Processing

Zpracování signálu - filtrace a vyhlazení dat.

- A_CORRELATE Compute the autocorrelation or autocovariance of a sample population.
- C_CORRELATE Compute the cross-correlation or cross-covariance of two sample populations.
- SMOOTH Smooth a time-series using a moving average (boxcar average).
- MEDIAN Median function and filter for time-series cleaning.
- FFT Fast Fourier transformation.
- HANNING Hanning window function.
- HAMMING Hamming window function.
- HILBERT Hilbert transformation (time-domain to time-domain).
- WTN Discrete wavelet transformation
- BLK_CON Discrete convolution an input signal with an impulse-response sequence.
- CONVOL Discrete convolution of two signals.
- DIGITAL_FILTER Computes impulse response of FIR filter based on Kaiser's window
- LEEFILT Performs the Lee filter algorithm on a vector a box of size $2N+1$.

Routines for Image Processing

Zpracování obrazu.

- SMOOTH Smooth a time-series using a moving average.
- FFT Fast Fourier transformation.

- Ověřování modelů polohy magnetopausy¹³ (ZS 2000/2001, autor: *Štefan Dušík*)
- Testování stripových polovodičových detektorů¹⁴ (ZS 2000/2001, autor: *Pavel Řezníček*)
- Analýza fluktuací metodou DFA (Detrended fluctuation analysis)¹⁵ (ZS 2001/2002, autor: *Martin Ondráček*)

Definice procedur a funkcí v IDL

Funkce explicitně vrací hodnotu, procedura může předat výsledek jen prostřednictvím parametrů.

Proměnné uvnitř procedury/funkce

- parametry funkce/procedury (poziční a klíčové)
- systémové proměnné (!)
- proměnné z bloku **COMMON**
`COMMON jmeno_bloku, var1, var2, ...`
- lokální (všechny ostatní proměnné považovány za lokální)

Typy parametrů

- poziční
 - povinné
 - nepovinné (aktuální počet přes `N_PARAMS()`, parametry lze ubírat jen zprava anebo zleva)
`pro` `nazev_procedury, par1, par2, ...`
- klíčové
 - klíčová slova (nezavislý odkaz přes jméno klíčového parametru)
`pro` `nazev_procedury, kpar_jmeno=kpar_odkaz`

¹³ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/dusik/zapocet.htm

¹⁴ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/reznicek/detektor.htm

¹⁵ http://aurora.troja.mff.cuni.cz/texty/IDL_kurz1/ukazky/ondracek/dfa.htm

Příklady

:define procedure s jedním povinným parametrem

```
pro FCEX1, x
  print, 'sin(',x,')=',sin(x)
end
```

:volani

```
FCEX1,!pi
```

:define procedure se dvěma povinnými parametry

```
pro FCEX2, x, y
  print, 'sin(',x,')=',sin(x)
  print, 'cos(',y,')=',cos(y)
end
```

:volani

```
FCEX2,!pi,!pi/4
```

:define procedure s dvěma nepovinnými parametry

```
pro FCEX3, x, y
  if N_PARAMS() eq 0 then return
  print, 'sin(',x,')=',sin(x)
  if N_PARAMS() gt 1 then print, 'cos(',y,')=',cos(y)
end
```

:volani

```
FCEX3,!pi,!pi/4
FCEX3,0
FCEX3
```

:define procedure s klicovým parametrem

```
pro FCEX4, x, SINX=s
  if keyword_set(s) then print, 'sin(',x,')=',sin(x)
end
```

:volani

```
FCEX4,!pi
FCEX4,!pi,/S
FCEX4,!pi,/SI
FCEX4,!pi,/SIN
FCEX4,!pi,/SINX
FCEX4,!pi,SI=1
```

:define funkce s klicovými parametry

```
function FCEX5, x, SINX=s, COSX=c
  if keyword_set(s) then return,sin(x)
  if keyword_set(c) then return,cos(x)
  return,!values.f_nan
```

:volani

Routines for Solving Nonlinear Equations

Řešení nelineárních rovnic.

- BROYDEN Solve sets of non-linear equations using a globally-convergent Broyden's method.
- FX_ROOT Compute the real and complex roots of a univariate non-linear function using Müller's method.
- FZ_ROOTS Compute the roots of a complex polynomial.
- NEWTON Solve sets of non-linear equations using a globally-convergent Newton's method.

Routines for Optimization

Optimalizační problémy.

- AMOEBA Multidimensional minimization of a user supplied function using the downhill simplex method.
- CONSTRAINED_MIN Solves nonlinear optimization problems.
- DFPMIN Davidon-Fletcher-Powell minimization of a user supplied function.
- POWELL Powell minimization of a user supplied function.

Routines for Multivariate Analysis

Multivarietní analýza.

- CLUST_WTS Compute the cluster weights of a multivariate data set.
- CLUSTER Compute a cluster analysis classification of a multivariate data set.
- CORRELATE Compute the linear correlation coefficient.
- CTI_TEST Construct contingency table from observed frequency data.
- KW_TEST Test the hypothesis that three or more sample populations have the same mean of distribution.
- M_CORRELATE Compute the multiple correlation coefficient.
- P_CORRELATE Compute the partial correlation coefficient.
- PCOMP Compute the principal components and derived variables of a multivariate data set.
- STANDARDIZE Compute standardized variables.

Routines for Handling Sparse Arrays

Zpracování řídkých matic.

Note that SPRSIN must be used to convert to sparse storage format before the other routines can be used.

- FULSTR Restore a row-indexed sparse array to full storage format.
- LINBCG Solve a system of linear equations using the iterative biconjugate method.

Routines for Derivation (Differentiation)

Algoritmy pro numerickou derivaci.

- DERIV Performs numerical differentiation using 3-point, Lagrangian interpolation and returns the derivative.
- DERIVSIG Computes the standard deviation of a derivative as found by the DERIV function, using the input variables of DERIV and the standard deviations of those input variables.

Routines for Integration

Numerická integrace funkcí a tabelovaných dat.

- CRVLENGTH Compute the length of a curve with tabular representation.
- INT_2D Evaluate the double integral of a bivariate function $f(x, y)$.
- INT_3D Evaluate the triple integral of a trivariate function $f(x, y, z)$.
- INT_TABULATED Integrate a tabulated data set $\{x_i, y_i = f(x_i)\}$.
- QROMB Evaluate integral over a closed interval using Romberg's method.
- QROMO Evaluate integral over an open interval using a modified Romberg's method.
- QSIMP Evaluate integral over a closed interval using Simpson's method.

Routines for Solving Simultaneous Linear Equations

Řešení soustav lineárních rovnic včetně přeурčených a nedourčených soustav.

- CHOLDC Construct the Cholesky decomposition of an array.
- CHOLSOL Solve sets of linear equations (use with CHOLDC).
- COND Compute the condition number of a square array.
- CRAMER Solve a linear system using Cramer's rule.
- DETERM Compute the determinant of a square array.
- GS_ITER Solve a linear system using Gauss-Seidel iteration.
- IDENTITY Create an identity array.
- INVERT Invert a square array.
- LU_COMPLEX Solve a complex linear system or invert a complex array.
- LUDC Construct the LU Decomposition of an array.
- LUMPROVE Iteratively improve the solution vector of a set of linear equations.
- LUSOL Solve sets of linear equations (use with LUDC).
- NORM Compute the infinity norm of a square array or the Euclidean norm of a vector.
- SVDC Construct the Singular Value Decomposition of an array.
- SVSOL Use back-substitution to solve a set of simultaneous linear equations (use with SVDC).
- TRACE Compute the trace of an array.
- TRISOL Solve a tridiagonal system of linear equations.

```
print, FCEX5(!pi)
print, FCEX5(!pi, /SINX)
print, FCEX5(!pi, /COSX)
print, FCEX5(!pi, /C, S=1) ;vrati sin(x)
```

Předávání dat hodnotou a referencí

```
;procedura menici hodnotu formálního parametru
pro H1, x
  print, x
  x=0
  print, x
end

;hodnotou
H1, 8

;hodnotou
x=2
H1, 2*x
print, x

;odkazem
x=2
H1, x
print, x
```

Práce s řetězci

Nejčastější procedury a funkce IDL pro práci s řetězci jsou (neuvádíme všechny možné klíčové parametry):

- vytvoření pole prázdných řetězců - funkce **Strarr(d1, ... dn)**
;příklad - pole 1x10 a 2x3
AA= Strarr(10)
BB= Strarr(2,3)
- vytvoření pole prázdných řetězců - funkce **Sindgen(d1, ... dn)**
;příklad - pole řetězců
['0', '1', '2', '3', '4', '5', '6', '7']
CC= Sindgen(8)

- převod výrazu do řetězcové proměnné - funkce **String(expr1, ... exprn, [Format='...'])**

;příklady

A= **String**(data)

B= **String**(hh, mm, ss, Format='(I2, ":", I2, ":", I2)')

;vektor bytových ciseł primo koduje retezec (zde "Hello")

C= **String**([72B, 101B, 108B, 108B, 111B])

- spojení řetězců - operátor +

;příklad

D= A + B

- spojení prvků pole řetězců - funkce **Strjoin**

;příklad

E= **Strjoin**(Sindgen(4))

- délka řetězce - funkce **Strlen**

- vyhledání podřetězce - funkce **Strpos**

;příklad

i= **Strpos**(B, 'i:')

- výběr podřetězce - funkce **Strmid**

;nalezení přípony souboru

j= **Strpos**(jmeno_souboru, '._', /Reverse_search)

pripona= **Strmid**(jmeno_souboru, j+1, **Strlen**(jmeno_souboru)-j)

- vložení řetězce **Strput**

;příklad

Strput, 'jedna tri', 'dva ', 6

- odstranění úvodních a koncových mezer - funkce **Strtrim**

;zepředu

Str= **Strtrim**('___ slovo')

;zezadu

Str= **Strtrim**('slovo___',1)

;z obou stran

Str= **Strtrim**('___ slovo ___', 2)

- převod na malá/velká písmena - funkce **Strlowercase/Struppercase**

- rozdělení řetězce - funkce **Strsplit...**, podporovány jsou i regulární výrazy

;nahrazení mezer pomlčkami

Str = 'Toto je veta.'

print, Strjoin(**Strsplit**(Str, /EXTRACT), '_')

;pozice znaku '/'

slash= **Strsplit**(cesta_k_souboru, '/')

- **SPL_INTERP** Compute cubic-spline interpolated values (use with **SPL_INIT**).
- **SPLINE** Cubic spline interpolation.
- **SPLINE_P** Parametric cubic spline interpolation.
- **TRIANGULATE** Construct a Delaunay triangulation of a planar set of points. With **TRIGRID**, this procedure can interpolate irregularly-gridded data to a regular grid.
- **TRIGRID** Compute a regular grid of interpolated values.
- **TRI_SURF** Compute a regularly- or irregularly-gridded set of points with a smooth quintic surface.
- **VORONOI** Compute the Voronoi polygon of a point.
- **WARP_TRI** Image array with a specified geometric correction applied.

Statistical Routines

Statistické výpočty.

- **HISTOGRAM** - histogram, hustotní funkce
- **HIST_2D** - bivarietní histogram
- **MOMENT** - Computes the mean, variance, skewness, and kurtosis, mean absolute and standard deviation of a sample population contained in an n-element vector X.
- **MAX, MIN** - minimum a maximum vektoru
- **MEAN, VARIANCE, SKEWNESS, KURTOSIS**
- **STDDEV, MEANABSDEV** - standardní odchylka od průměru, střední absolutní odchylka od průměru nebo mediánu
- **MEDIAN** - medián vektoru

Routines for Hypothesis Testing

Statistické testování hypotéz o vztahu a vlastnostech souborů dat.

- **CTI_TEST** Construct contingency table from observed frequency data.
- **FV_TEST** Compute the F-statistic for two sample populations.
- **KW_TEST** Test the hypothesis that three or more sample populations have the same mean of distribution.
- **LNP_TEST** Compute the Lomb Normalized Periodogram of two sample populations.
- **MD_TEST** Test the hypothesis that a sample population is random.
- **R_CORRELATE** Compute the rank correlation of two sample populations.
- **R_TEST** Test the hypothesis that a binary population is random.
- **RS_TEST** Test the hypothesis that two sample populations have the same mean of distribution.
- **S_TEST** Test the hypothesis that two sample populations have the same mean of distribution.
- **TM_TEST** Compute the student's t-statistic for two sample populations.
- **XSQ_TEST** Compute the Chi-square goodness-of-fit between observed and expected frequencies.

Routines for Curve and Surface Fitting

Prokládání křivek a ploch.

- COMFIT Gradient-expansion least squares fit to paired data.
- CRVLENGTH Compute the length of a curve with tabular representation.
- CURVEFIT Non-linear least squares fit to a function.
- GAUSSFIT Fit sum of a gaussian and a quadratic.
- LADFIT Least absolute deviation fit to paired data.
- LINFIT Minimal Chi-square fit to paired data.
- POLY_FIT Polynomial least squares fit.
- POLYFITW Weighted polynomial least squares fit.
- REGRESS Multiple linear regression.
- SFIT Determine a polynomial fit to a surface.
- SVDFIT General least squares fit using SVD.

Routines for Computing Eigenvalues and Eigenvectors

Hledání vlastních čísel a vlastních vektorů matic.

- EIGENQL Compute eigenvectors of a real, symmetric array, given the array.
- EIGENVEC Compute eigenvectors of a real, nonsymmetric array, given the array and its eigenvalues.
- ELMHES Reduce a real, nonsymmetric array to upper-Hessenberg form.
- HQR Compute the eigenvalues of an upper-Hessenberg array.
- TRIQL Compute eigenvalues and eigenvectors of a real, symmetric, tridiagonal array.
- TRIRED Use Householder's method to reduce a real, symmetric array to tridiagonal form.

Routines for Gridding and Interpolation

Interpolace, změna mříže nezávislých hodnot.

- BILINEAR Bilinear interpolation.
- GRID3 Smooth fit to a set of 3D scattered nodes.
- INTERPOL Linear interpolation of vectors.
- INTERPOLATE Compute linear, bilinear, or trilinear interpolates.
- KRIG2D Interpolate regularly or irregularly gridded points using kriging.
- MIN_CURVE_SURF Interpolate regularly or irregularly gridded points using minimum curvature spline surface.
- POLAR_SURFACE Interpolate a surface from polar coordinates to rectangular coordinates.
- SPH_CUT Spherical gridding. Scattered samples on the surface of a sphere are interpolated to a regular grid.
- SPL_INIT Establish interpolating spline for a data set (use with SPL_INTERP).

- porovnání řetězců - funkce **Strcmp** nebo operátory **eq/ne**

;příklady

```
if A eq B then ...
```

```
if Strcmp(A, B) then ...
```

;porovnání prvních n znaků bez ohledu na velika/mala písmena

```
if Strcmp(A, B, n, /Fold_case) then ...
```

- porovnání řetězců s možností *wildcardů* - funkce **Strmatch**

;nalezení všech 4-znakových slov začínajících "f" nebo "F" a končících "t" or "T":

```
str = ['foot', 'Feet', 'fate', 'FAST', 'ferret', 'fort']
```

```
Print, str[Where(Strmatch(str, 'f?t', /Fold_case) eq 1)]
```

Procedury pro práci se soubory

Otevírání souborů

- otevření souborů pro čtení **OpenR, unit, filename, [/GetLUN], [/Compress], ...**

;unit je číslo jedinečné mezi otevřenými soubory

;/GetLUN zajistí jeho nalezení (>100)

```
OpenR, 2, "soubor.dat"
```

```
OpenR, unit_variable, filename_string, /GetLUN
```

; lze též použít proceduru GetLUN

```
GetLUN, unit1 & OpenR, unit1, filename_string
```

;/Compress komprimuje soubor jako GZIP

```
OpenR, 2, "soubor.gz", /Compress
```

- otevření souboru pro zápis **OpenW, unit, filename, [/GetLUN], [/Append], [/Compress], ...**

;/Append otevře soubor na konci

```
OpenW, 2, "novy_soubor.txt"
```

```
OpenW, 2, "stary_soubor.txt", /Append
```

- otevření pro čtení i zápis (opravy) **OpenU, unit, filename, [/GetLUN], [/Append], [/Compress], ...**

```
OpenU, 2, "upravovany_soubor.txt"
```

Uzavření souborů

- uzavření **Close**, [**unit1**, ...**unitn**,] [/All]
 - : lze uzavřít jeden i více souborů
 - : /All uzavře všechny
 - Close, 2
 - Close, unit2, unit3, unit4
 - Close, /All
- uzavření s uvolněním unit **Free_LUN**, **unit1**, ...**unitn**
 - : lze uzavřít jeden i více souborů
 - Free_LUN, unit1
 - Free_LUN, unit2, unit3, unit4

Čtení textových dat

- vstup z příkazové řádky **Read**, [**prompt**], **var1**, ...**varn**
- vstup ze souboru **ReadF**, **unit**, **var1**, ...**varn**
- vstup z řetězce **ReadS**, **input_string**, **var1**, ...**varn**
 - : příklady*
 - Read, 'Zadej delku:', delka
 - ReadF, unita, x,y,z
 - ReadS, mystring, data
 - : formátované čtení*
 - a=F1tarr(10, /Nozero) & ReadF, 5, a, b, c, Format='(10F8.3, 2X, I4, 2X, G10.5)'

Zápis textových dat

- do output logu **Print**, **expr1**, ...**exprn**
- do souboru **PrintF**, **unit**, **expr1**, ...**exprn**
 - do řetězcové proměnné - funkce **String**(**expr1**, ...**exprn**, [Format='...'])
 - : příklady*
 - Print, 'Delka:', delka
 - PrintF, unitb, x, ' - ', y, z
 - a=String(vyska, ' cm')
 - : formátovaný zápis*
 - PrintF, 2, a, b, Format='("delka ", F8.3, "vyska ", F5.2)'

Přehled metod změny velikosti pole (obrazu)			
Funkce	zvětšení pole	zmenšení pole	pozn.
CONGRID()	vzorkování nejbližších sousedů (1D, 2D) lineární interpolace (3D)		1D, 2D, 3D pole
CONGRID(CUBIC=x)	interpolace parametrickou kubickou konvolucí		1D, 2D pole
CONGRID(/INTERP)	lineární interpolace		1D, 2D, 3D pole
EXPAND()	bilineární interpolace	bilineární interpolace	jen 2D pole

Routines for Random Number and Distribution Generation, Probability Computation

Generování pseudonáhodných čísel a rozdělení. Výpočty pravděpodobnosti.

- RANDOMN The Box-Muller method for generating normally-distributed (Gaussian) random numbers.
- RANDOMU Returns one or more uniformly-distributed, floating-point, (also binomial, gamma, normal, Poisson distribution) pseudo-random numbers.
- BINOMIAL, CHISQR_PDF, T_PDF, F_PDF, GAUSS_PDF - pravděpodobnost $P(X>V)$ pro náhodnou veličinu X s kumulativním binomickým, chí-kvadrát, Studentovým, F a normálním Gaussovým rozdělením
- CHISQR_TCF, T_CVF, F_TCV, GAUSS_TCV - určení hraniční hodnoty V, kde $P(X>V)=p$ pro náhodnou veličinu X s chí-kvadrát, Studentovým, F a normálním Gaussovým rozdělením

Routines for Computing Correlations

Výpočty korelací.

- A_CORRELATE Compute the autocorrelation or autocovariance of a sample population as a function of the lag.
- C_CORRELATE Compute the cross-correlation or cross-covariance of a sample population as a function of the lag.
- CORRELATE Compute the linear correlation coefficient.
- M_CORRELATE Compute the multiple correlation coefficient.
- P_CORRELATE Compute the partial correlation coefficient.
- R_CORRELATE Compute the rank correlation of two populations.
- RANK Compute the magnitude-based ranks of a sample population X.

Manipulace s poli:

- INVERT - inverze čtvercové matice (pole)
- REFORM - změna počtu a velikosti dimenzí pole beze změny jeho obsahu (hodnot prvků)
- REVERSE - změna pořadí prvků vektoru nebo pole
- ROTATE - rotace prvků 2D pole o 0°, 90°, 180°, 270° s možnou transpozicí
- ROT - rotace obrazu (2D pole) o libovolný úhel (výpočet nových prvků interpolací)
- SHIFT - posun prvků pole
- SORT - setřídění prvků pole dle velikosti s návratem odpovídajícího pole indexů
- TRANSPOSE - transpozice pole
- CROSSP - vektorový součin tříprvkových vektorů

Další rutiny pro práci s poli a obrazy:

- MIN, MAX - nalezení nejmenšího a největšího prvku pole
- MEDIAN - nalezení mediánu pole nebo mediánový filtr
- SIZE - údaje o velikosti a typu pole
- UNIQ - indexy jedinečných prvků pole
- WHERE - nalezení indexů nenulových prvků pole
- REPLICATE_INPLACE - nahrazuje prvky vybrané části pole určitou hodnotou
- BLAS_AXPY - přičítá ke stávajícímu poli nebo jeho části násobek jiného pole
- VALUE_LOCATE - hledá intervaly v daném monotoním vektoru, které ohraničují vyhledávané hodnoty

Změna velikosti pole:

- REFORM - změna počtu a velikosti dimenzí pole beze změny jeho obsahu (hodnot prvků)
- TOTAL - částečný nebo úplný součet prvků pole
- CONGRID - změna velikosti pole (obrazu)
- REBIN - změna velikosti pole (obrazu) v celočíselných násobcích
- EXPAND - změna velikosti pole (obrazu)

Přehled metod změny velikosti pole (obrazu)			
Funkce	zvětšení pole	zmenšení pole	pozn.
REBIN()	bilineární interpolace	průměr z okolí bodu	1-8D pole, jen celočíselné násobky/podíly velikosti původní a nové dimenze
REBIN(/SAMPLE)	vzorkování nejbližších sousedů	vzorkování nejbližších sousedů	1-8D pole, jen celočíselné násobky/podíly velikosti původní a nové dimenze

Čtení a zápis binárních dat

- vstup ze souboru **ReadU, unit, var1, ...varn**
- zápis do souboru **WriteU, unit, var1, ...varn**

:příklady - při čtení se čte tolik dat, kolik požaduje typ dane promenne
`x=0 & mm=Db1arr(10,5, /Nozero) & ReadU, 1, x,mm`
`WriteU,_unit_out, data`

Utility

- vyprázdnění bufferů **Flush, unit1, ...unitn**

`Flush, unit3`

- status souboru - funkce **Fstat(unit)** vrací *FSTAT* strukturu

:příklad
`A = Fstat(1)`
:napr. tisk jména, délky, aktuální pozice v souboru:
`Print, 'File: ', A.NAME, ' Length: ', A.SIZE, ' Pos: ', A.CUR_PTR`

- ošetření chyb **On_IOerror, label**
- nalezení všech souborů podle masky - funkce **FindFile(filespec,[count=variable])**
- změna pracovního adresáře **PushD, directory / PopD**

`On_IOerror, iochyba`
`PushD, 'C:\temp'`
:nalezne jmena vsech souboru C:\temp.tmp*
`a=FindFile('*.*.tmp',count=ca)`
`PopD`
`...`
`iochyba:`
:zde osetreni chyb

Dialogy a makra

- výběr souboru - funkce **Dialog_pickfile**

:příklad - otevreme soubor pro vstup dat
`fn = Dialog_Pickfile(file='nejcastejsi.dat', path='cesta k souborum',`
`/read, title='Vyber soubor pro cteni', filter='*.dat')`
`if fn ne '' then begin`
`OpenR, u, fn, /Get_LUN`
`....`

```

;otevreme soubor pro tiskovy vystup
fn = Dialog_Pickfile(file='image.ps', path='C:\temp\', /write,
    title='Zadej soubor pro vystup v PostScriptu', filter='*.ps')
Device, filename=fn & Set_plot, 'PS'

```

- makro pro načtení textového souboru - funkce **Read_ASCII**
- makro pro načtení binárního souboru - funkce **Read_binary**

Další rutiny pro čtení textových a binárních souborů

K rychlému čtení textových a binárních souborů v IDL lze použít v menu předdefinovaná makra (též tlačítka na pracovní liště) `IMPORT_ASCII_FILE` a `IMPORT_BINARY_FILE`. V dialogu zjišťují umístění souboru, vnitřní strukturu, oddělovače, názvy a datové typy polí atd. Jejich výsledkem je proměnná typu *struktura*, jejíž položky tvoří datová pole (např. u textových souborů jsou to jednotlivé sloupce).

K opakovanému čtení souborů téhož formátu je výhodné připravit obdobným dialogem pouze *template* a ten využít ve čtecích rutinách:

- vytvoření *template* – funkce **ASCII_template([filename])** a **Binary_template([filename])**

```

;příklady - nazada-li se jmeno souboru, otevře se nejprve dialog k
vyberu souboru

```

```

Atemp11= ASCII_template('c:\temp\soubor.txt')
Atemp12= ASCII_template()
Btemp1= Binary_template(Filepath('surface.dat',
    Subdir=['examples','data']))

```

Funkce `Filepath` slučuje jméno souboru a podadresáře s cestou k instalaci IDL (např. "C:\program files\rsi\idl53\"). Pro opakované použití *template* po opuštění IDL lze použít procedury `Save` a `Restore`. Ukládá a načítá se celá proměnná včetně jména:

```

;ulozeni template
Save, Atemp11, 'c:\temp\soubor_template.dat'
;obnoveni template (promenne Atemp11)
Restore, 'c:\temp\soubor_template.dat'

```

- načtení dat - funkce **Read_ASCII(filename, Template=templ)** a **Read_binary(filename, Template=templ)**

```

;příklady
Adata= Read_ASCII('c:\temp\soubor.txt', Template=Atemp11)
Bdata= Read_Binary('examples\data\surface.dat', Template=Btemp1)

```

Tyto dvě funkce také definují další klíčové parametry pro přímou definici formátu vstupního souboru - viz IDL help.

Routines for Mathematical Error Assessment

Testování statusu matematických chyb a strojových charakteristik (přesnost výpočtu).

- `CHECK_MATH` Return and clear accumulated math error status.
- `FINITE` Returns TRUE if argument is finite.
- `MACHAR` Returns machine-specific parameters that affect floating-point arithmetics.

Mathematical Functions

Různé speciální funkce.

- `ABS`, `ROUND`, `FLOOR`, `CEIL` - absolutní hodnota, zaokrouhlení, celá část
- `CONJ`, `COMPLEXROUND` - komplexně sdružené hodnoty, zaokrouhlení reálné a imaginární části komplexního čísla
- `SIN`, `COS`, `SINH`, `COSH`, `ASIN`, `ACOS`, `TAN`, `TANH`, `ATAN` - goniometrické, hyperbolicke a k nim inverzní funkce
- `SQRT`, `ALOG`, `ALOG10`, `EXP` - odmocnina, přirozený, desítkový logaritmus, exponenciála
- `EXPINT`, `ERRORF`- `En(x)`, `erf(x)`
- `FACTORIAL` - faktoriál nebo jeho Stirlingova aproximace
- `BESSELJ`, `BESSELK`, `BESSELN` - Besselovy funkce
- `BETA`, `GAMMA`, `IBETA`, `IGAMMA`, `LNGAMMA` - úplné a neúplné Beta a Gama funkce, logaritmus Gama funkce
- `POLY` - počítá hodnoty polynomiální funkce zadané koeficienty
- `PRIMES` - hledá prvních K prvočísel
- `VOIGT` - profil intenzity atomové absorpční čáry (VOIGT profile)

Routines for Array and Image Manipulation

Základní práce s poli a obrazy - vytváření polí:

- `BYTARR`, `COMPLEXARR`, `DBLARR`, `DCOMPLEXARR`, `FLTARR`, `INTARR`, `LON64ARR`, `LONARR`, `UINTARR`, `ULON64ARR`, `ULONARR` - vytváření polí nulových prvků v různých datových typech
- `OBJARR`, `PTRARR` - vytváření polí prázdných objektů a ukazatelů
- `STRARR` - vytváření polí prázdných řetězců
- `MAKE_ARRAY` - univerzální rutina pro vytvoření pole prvků požadovaného typu a počtu dimenzí
- `REPLICATE` - vytvoření pole prvků replikací hodnoty potřebného typu
- `BINDGEN`, `CINDGEN`, `DINDGEN`, `DCINDGEN`, `FINDGEN`, `INDGEN`, `L64INDGEN`, `LINDGEN`, `UINDGEN`, `UL64INDGEN`, `ULINDGEN`, `SINDGEN` - vytváření polí se vztupně číslovanými prvky v různých datových typech (hodnota prvku je rovna jeho jednodimensionálnímu indexu)
- `IDENTITY` - jednotkové matice
- `DIST` - 2D pole, hodnoty prvků úměrné frekvenci (vzdálenosti od nejbližšího rohu)

Matematické rutiny a algoritmy IDL

v5.3

Následuje stručný přehled procedur a funkcí IDL pro zpracování dat, seřazených dle kategorie jejich použití.

Distribuce IDL obsahuje celý soubor matematických rutin a numerických algoritmů nutných pro základní typy zpracování dat. Algoritmy jsou vešměs převzaty z *Numerical Recipes in C: The Art of Scientific Computing (Second Edition)*. Podrobný popis následujících rutin viz online **help** a manuály IDL (Reference Guide).

- Testování statusu matematických chyb a strojových charakteristik (přesnost výpočtu)
- Matematické funkce
- Základní rutiny pro práci s poli a obrazy
- Generování pseudonáhodných čísel a rozdělení, výpočty pravděpodobnosti
- Výpočty korelací
- Prokládání křivek a ploch
- Hledání vlastních čísel a vlastních vektorů matic
- Interpolace, změna mříže nezávislých hodnot
- Statistické výpočty
- Statistické testování hypotéz o vztahu a vlastnostech souborů dat.
- Algoritmy pro numerickou derivaci.
- Numerická integrace funkcí a tabelovaných dat
- Řešení soustav lineárních rovnic včetně přeurčených a nedourčených soustav
- Řešení nelineárních rovnic
- Optimalizační problémy
- Multivarietní analýza
- Zpracování řídkých matic
- Zpracování časových řad
- Zpracování signálu - filtrace a vyhlazení dat
- Zpracování obrazu
- Trasování vektorových polí
- Analytická geometrie
- Zpracování objemových dat

Grafický výstup, okna v IDL

Grafický výstup v IDL často směřujeme do jednoho nebo více grafických oken na obrazovce monitoru. Pod různými operačními systémy má toto zařízení (device) různé názvy:

- 'WIN' - MS Windows NT, 95/98/2000
- 'Z' - X Windows (různé varianty UNIX)
- 'MAC' - Macintosh/Power Mac

Vedle toho existují další výstupní zařízení (např. pro tiskárny a plotery):

- 'PS' - postscriptová tiskárna
- 'PCL' - výstup v PCL jazyce (napr. tiskárny Hewlett Packard)
- 'HP' - výstup ve formátu HPGL (napr. plotery)
- 'LJ' - barevné tiskárny DEC
- 'PRINTER' - grafické výstupní zařízení (ovladače) operačního systému
- 'Z' - virtuální výstupní zařízení v paměti, používá se např. k přípravě bitmap pro soubory v grafických formátech (JPG, PNG, GIF, BMP atd.) nebo pro 3D.

Přepínání mezi různými výstupními zařízeními se provádí příkazem **SET_PLOT**:

```
Set_plot, 'WIN'
```

Grafickým oknům přísluší určité atributy:

- *aktivní okno* - okno používané v dané chvíli IDL pro grafický výstup (viz !D.Window)
- *okno s fokusem* - okno vybrané uživatelem, zpravidla leží na vrcholu desktopu (nad všemi ostatními okny na obrazovce)

V souvislosti s obnovením obsahu okna předtím překrytého, IDL rozlišuje 3 módy (nastavení v Preferencích nebo klíčovým parametrem **RETAIN** procedur DEVICE nebo WINDOW):

- 0 - obsah okna se neuchovává, je nutné nové vyvolání grafických kreslicích rutin)
- 1 - Xserver nebo operační systém zajišťuje uložení bitmapy (*backing store*)
- 2 - backing store zajišťuje samo IDL

Otevírání nového okna

První okno se otevírá automaticky. Další otevíráme příkazem **WINDOW**. Nové okno je automaticky *aktivní*. Znovuotevřením okna s tímž číslem se jeho obsah smaže.

```
;otevření okna n=0..31  
;window je totožné s Window,0  
Window, n  
;okna z rozsahu 32..127, automatické přidelení volného čísla  
Window, /Free
```

Možné klíčové parametry jsou např.

- Title - titulek v liště
- Xpos, Ypos - souřadnice okna v pixelech
- Xsize, Ysize - velikost okna v pixelech

```
;příklad  
Window, /Free, Xsize=400, Ysize=400, Title='ctvercove okno'
```

Další operace s okny

- uzavření **Wdelete**, [okno1, ...]

```
;příklad  
Wdelete, 2  
;uzavre aktivní okno  
Wdelete
```

- výběr aktivního okna **Wset**, **okno**

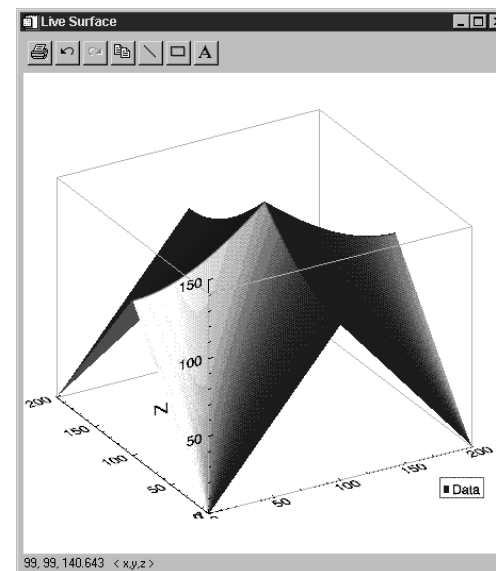
```
Wset, 4
```

- přepnutí fokusu nebo skrytí okna

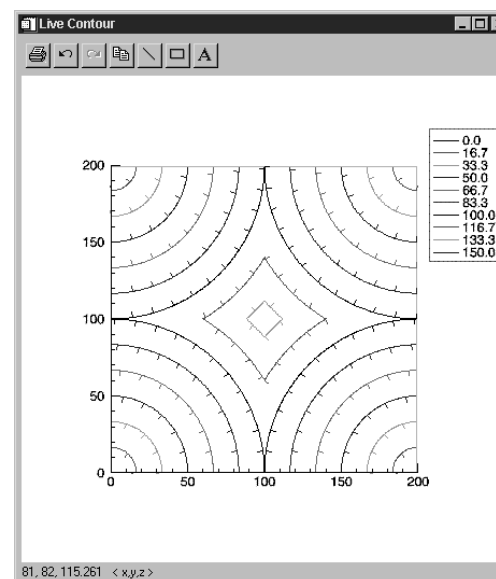
```
;přenesí okna navrch  
Wshow, 8  
;skrytí okna  
Wshow, 1, Show=0
```

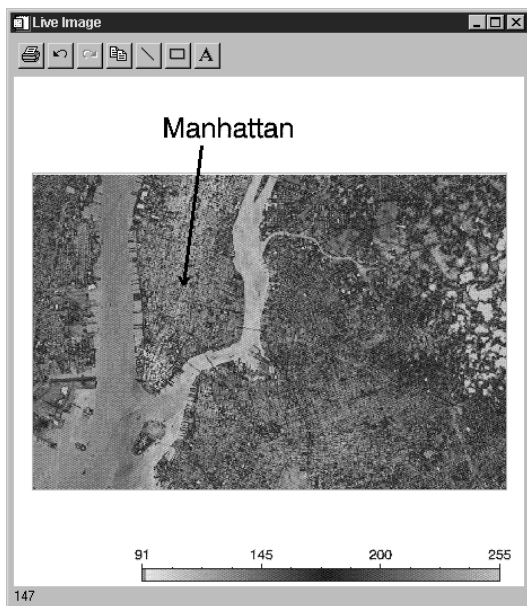
- smazání okna nebo přechod na novou stránku na tiskárně

```
;příklady  
Erase  
Erase, Background_color=barva_pro_pozadí
```



:zobrazení tehoz 2D pole - kontury
Live_contour, Dist(200)





Zobrazení 2D polí - LIVE_SURFACE a LIVE_CONTOUR

Dvoudimenzionální pole jako povrch v 3D prostoru nebo pomocí *izočar* (kontur) interaktivně zobrazují procedury LIVE_SURFACE a LIVE_CONTOUR. Interaktivně se volí způsob kreslení povrchu a jeho stínování, popis os, počet a vlastnosti kontur, další textové anotace, čáry, šipky a boxy v okně. Obsah okna lze opět přímo zaslat na systémovou tiskárnu a zkopírovat do *clipboardu*. Při úpravách vzhledu grafu lze použít funkce *Undo* a *Redo*. V levém dolním rohu okna se zobrazuje hodnota prvku pole, na který právě ukazuje kurzor myši, nebo nápověda k tlačítkům v nástrojové liště.

```
;zobrazení 2D pole generovaného funkcí Dist - 3D povrch
Live_surface. Dist(200)
```

Kreslení čárových grafů v IDL I

Základní použití příkazu PLOT

```
x=!pi*2/100*FINDGEN(100)
;data proti indexu
PLOT,sin(x)
;nezavisle a zavisle promenne
PLOT,x,sin(x)
;graf parametricky zadane fce
PLOT,cos(x),sin(x)
```

Klíčová slova pro PLOT (a další grafické procedury)

```
;popisy, titulky PLOT,x,sin(x), ...
TITLE='horni titulek'
SUBTITLE='spodni titulek'
TITLE='popis osy X'
YTITLE='popis osy Y'
CHARSIZE=relativni_velikost_pisma
XCHARSIZE=rel_velikost_popisu_X
YCHARSIZE=rel_velikost_popisu_Y
;priklad
PLOT,x,sin(x), TITLE='horni',SUBTITLE='spodni', XTITLE='popis osy X',
YTITLE='popis osy Y', CHARSIZE=2, XCHARSIZE=0.5, YCHARSIZE=1.2
;barvy
COLOR=barva_popredi
BACKGROUND=barva_pozadi
;nastaveni barevne skały "duha" (indexy barev 0 az !D.table_size-1)
DEVICE, DECOMPOSED=0 & LOADCT,39
PLOT,x,sin(x),COLOR=200
;rozsah os
/YNOZERO ;osa Y nezacina v 0
XRANGE=[xmin,xmax]
YRANGE=[ymin,ymax]
PLOT,x,sin(x)+3
PLOT,x,sin(x)+3,/YNOZERO
PLOT,x,sin(x),XRANGE=[0,10],YRANGE=[-2,2]
;rozsah zobrazenych dat
MIN_VALUE,MAX_VALUE
PLOT,x,sin(x),MIN_VALUE=0.25,MAX_VALUE=0.75
;umistení grafu
POSITION=[x_vlevo_dole,y_vlevo_dole,x_vpravo_nahore,y_vpravo_nahore]
/NORMAL ;normalni souradnice okna [0..1,0..1]
```

```

/DEVICE ;souradnice zarizeni napr. [0..640,0..480]
/DATA ;datova transformace pro definovane osy = box [0..1,0..1]
;styl os
XSTYLE, YSTYLE= soucet masek stylu

```

1 - přesný rozsah osy, 2 - rozšíření rozsahu, 4 - osa se nekreslí, 8 - osa se kreslí jen dole (vlevo), 16 - osa Y nezačíná v 0 (jako /YNOZERO)

```

;styl car grafu
LINESTYLE= styl

```

0 - souvislá, 1 - tečkovaná, 2 - čárkovaná, 3 - čerchovaná, 4 - čárka + tři tečky, 5 - dlouhé čárky

```

;styl bodu grafu (PSYM >0 ... kreslí se jen body, PSYM <0 ... kreslí se
body i čáry)
PSYM= styl

```

1-7 plus, hvězdička, tečka, kosočtverec, trojúhelník, čtverec, 8 uživatelský symbol (definovaný přes USERSYM), 10 mod histogram

Anotace příkazem XYOUTS

```
XYOUTS, 0.5,0.8, 'dalsi popis'
```

Kreslení čárových grafů v IDL II

Příkaz PLOT - stupnice na osách

logaritmická osa X:

```
PLOT, x, y, /XLOG
```

logaritmická osa Y:

```
PLOT, x, y, /YLOG
```

logaritmická osa X a Y:

```
PLOT, x, y, /XLOG, /YLOG
```

Příkaz PLOT - redukce bodů grafu

zobrazení průměru n bodů:

```
PLOT, x, y, NSUM=n
```

Příkaz PLOT - koordináty

polární souřadnice - box:

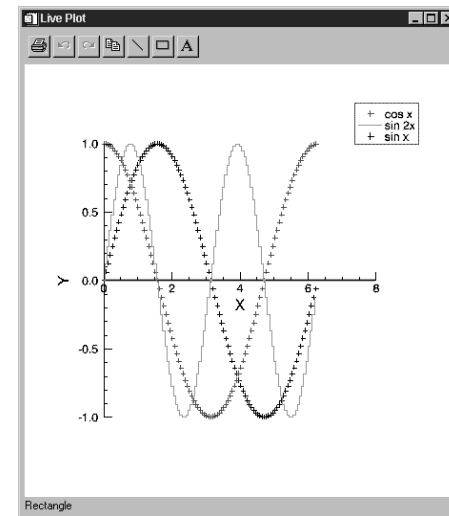
```
PLOT, r, phi, /POLAR
```

polární souřadnice - kříž (potlačit kreslení os a použít příkaz AXIS):

:jiné typy grafu

```
live_plot, sin(x), cos(x), name={data:['sin x', 'cos x']}, independent=x,
/scatter
```

```
live_oplot, sin(2*x), name={data:['sin 2x']}, independent=x,
subtype='histogram'
```



Zobrazení 2D polí - bitmapy s LIVE_IMAGE

Dvoudimenzionální pole jako bitmapy interaktivně zobrazuje procedura LIVE_IMAGE. Interaktivně lze zvolit použitou barevnou tabulku a nastavit barevnou škálu a její popis, další textové anotace, čáry, šipky a boxy v okně. Obsah okna lze opět přímo zaslat na systémovou tiskárnu a zkopírovat do *clipboardu*. Při úpravách vzhledu grafu lze použít funkce *Undo* a *Redo*. V levém dolním rohu okna se zobrazuje hodnota pixelu, na který právě ukazuje kurzor myši, nebo nápověda k tlačítkům v nástrojové liště.

:načtení a zobrazení bitmapy v barevné tabulce "Haze"

```
image=Read_tiff(Filepath('image.tif', Subdir=['examples', 'data']))
```

```
Help, Image
```

```
IMAGE BYTE = Array[768, 512]}
```

```
Live_image, image
```

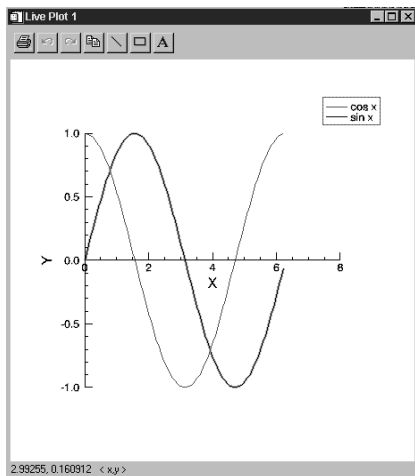

zobrazují souřadnice bodu vybrané křivky, na který právě ukazuje kurzor myši, nebo nápověda k tlačítkům v nástrojové liště.

Níže uvedené příklady nezahrnují úplný výčet všech klíčových parametrů - viz dokumentace IDL.

;příklad - základní interaktivní okno

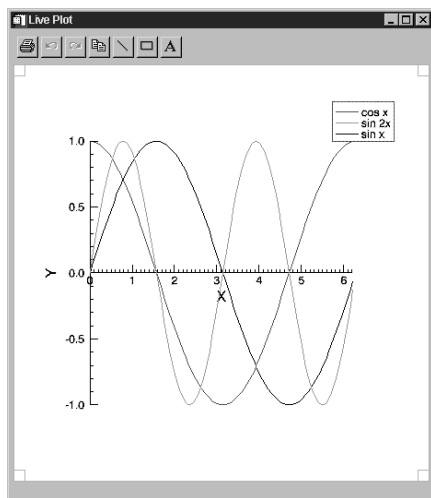
`x=!pi*2/100*findgen(100)`

`live_plot,sin(x),cos(x), name={data:['sin x','cos x']}, independent=x`



;přidání další křivky do tohoto okna

`live_oplot,sin(2*x), name={data:['sin 2x']}, independent=x`



`PLOT,r,phi,/POLAR,XSTYLE=4,YSTYLE=4`

`AXIS,0,0,/XAXIS`

`AXIS,0,0,/YAXIS`

Příkaz PLOT - časová osa

práce s časovou osou s využitím procedur JULAX,JULNORM (ke stažení u autora):

;simul.data

`x=RANDOMU(seed,100)`

;pocateční čas... JULTIME(ms,sec,min,day,month,year)

`t0=JULTIME(0,0,0,7,11,2000)`

;vektor času po 1 minutě (1 den = 1440 min)

`t=t0+DINDGEN(100)*1D/1440`

;rozsah času [t(0),t(N_ELEMENTS(t)-1)]

`tr=[MIN(t),MAX(t)]`

;definice časové osy

`JULAX,tr,ts,tv,tn,tm,NINT=6`

;kreslení dat a nastavení časové osy

`PLOT,JULNORM(t,tr),$;normalizace času t do rozsahu 0..1`

`X,$;vektor dat`

`XTICKS=ts,$;počet hlavních tiků`

`XTICKV=tv,$;polohy hlavních tiků`

`XTICKN=tn,$;popis hlavních tiků`

`XMINOR=tm ;počet malých tiků`

Umístění grafů v okně - proměnná !P.MULTI

Více grafů v jednom okně

`!P.MULTI=[zbyvajici_pocet_v_okne,pocet_sloupců,pocet_řad,pocet_v_Z,{0-po_sloupcích,1-po_řadách}]`

Alternativně lze zadat jen hodnoty prvních tří položek.

Příklad 1.

;nastavení rozvržení grafu v okně

`!P.MULTI=[0,2,3,1,1]`

;zvětšíme velikost písma

`!P.charsize=2`

;nezavislá proměnná

`x=2*!pi/100*findgen(100)`

;kreslíme grafy

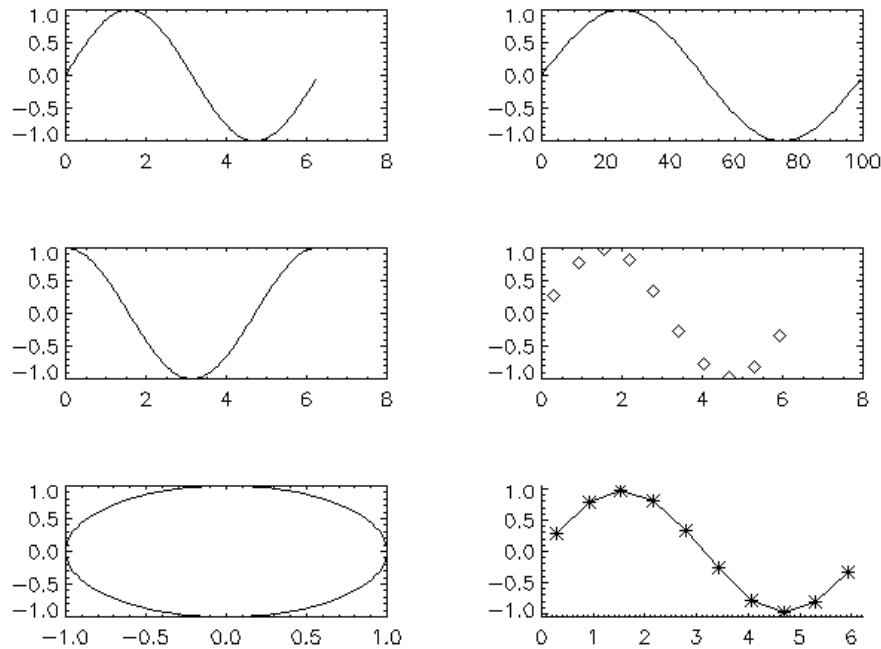
`plot,x,sin(x)`

`plot,x,cos(x)`

```

plot,cos(x),sin(x)
plot,sin(x)
plot,x,sin(x), psym=4, nsum=10
plot,x,sin(x), psym=-2, xstyle=9, ystyle=10, nsum=10

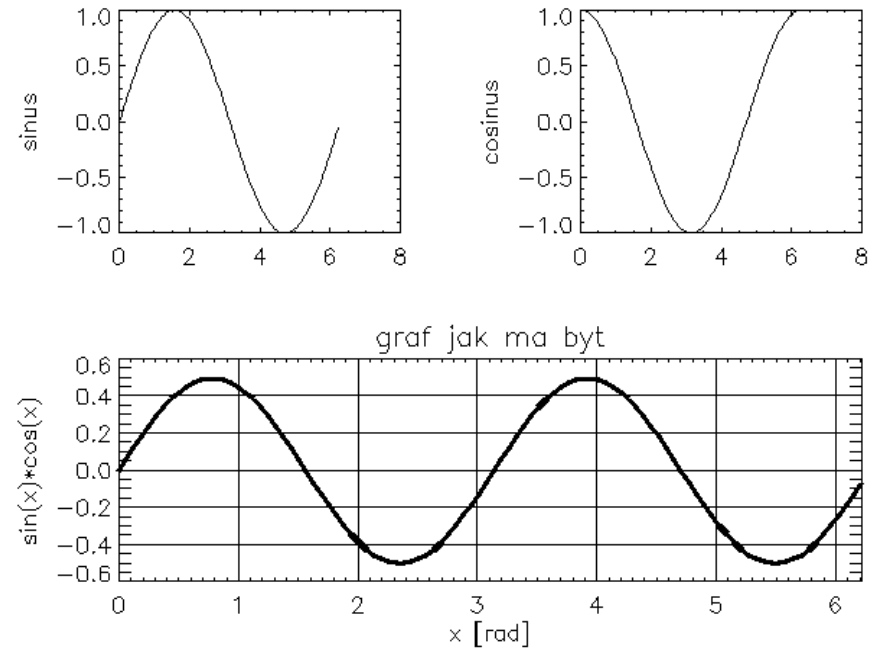
```



```

xstyle=1, xtitle='x [rad]', ytitle='sin(x)*cos(x)', $
thick=3, ticklen=1

```



Zpět jeden graf

```
!P.MULTI=0
```

Příklad 2.

```

;nastaveni rozvrzeni grafu v okne
;variantou je !P.MULTI=[0,2,2,1,0]
!P.MULTI=[0,2,2]

```

```

;nazavisla promenna
x=2*!pi/100*findgen(100)

```

```

;kreslime grafy
plot,x,sin(x), ytitle='sinus'
plot,x,cos(x), ytitle='cosinus'

```

```

;nastaveni rozvrzeni grafu v okne - spodni polovina
!P.MULTI=[1,1,2]

```

```

plot,x, cos(x)*sin(x), title='graf jak ma byt', $

```

Interaktivní rutiny pro 2D a 3D grafiku

V distribuci IDL je obsaženo několik rutin, které umožňují interaktivní úpravy nakresleného grafu v grafickém okně. Tyto rutiny lze volat samostatně, ale data s jejich pomocí zobrazuje i aplikace **INSIGHT** spouštěná pod IDL.

Čarové grafy - LIVE_PLOT a LIVE_OPLOT

Tyto procedury se hodí ke kreslení čarových a bodových grafů a histogramů. Interaktivně lze nastavit škálu a popis os, atributy jednotlivých křivek, titulky a další textové anotace, čáry, šipky a boxy v grafu. Obsah okna lze přímo zaslat na systémovou tiskárnu nebo zkopírovat do *clipboardu*. Při úpravách vzhledu grafu je možné také použít funkce *Undo* a *Redo*. V levém dolním rohu okna se