INTRO

- This is a program for χ^2 fitting, using SN1a data.
- SN data are taken from http://supernova.lbl.gov/Union/
-
-
-

PRE

Settings

Include the packages needed.

```
Needs["ErrorBarPlots`"]
<< PhysicalConstants`
```

Set work directory. Modify it to your own directory before runing this program. DATA folder are placed in this directory. DATA folder should contain the SN data files.

```
SetDirectory[
  "E:\\Nutshare\\Store\\Projects\\Growth\\git\\CoChiSquare"];
```

Conventions, parameters, etc

Line element: $ds^2 = -dt^2 + a[t]^2 \left(\frac{dr^2}{1-k\,r^2} + r^2\left(d\theta^2 + \mathrm{Sin}[\theta]^2\, d\phi^2\right)\right)$

Ωm0: matter fraction
Ωd0: dark energy/cosmological constant fraction
Ωr0: radiation fraction

Ωm0w: matter fraction given by WMAP
Ωd0w: dark energy/cosmological constant fraction given by WMAP
Ωr0w: radiation fraction given WMAP

```
Ωm0w = 0.27;
Ωd0w = 0.73;
Ωr0w = 8.09 * 10^-5;
```

Basic Equations in LCDM

H0: Hubble constant in unit of km/s/Mpc
c: speed of light

```
H0 = 71;
c = SpeedOfLight * Second/(1000 Meter)
gra = GravitationalConstant
```

$$\frac{149896229}{500}$$

$$\frac{6.67428\times 10^{-11}\ \text{Meter}^2\ \text{Newton}}{\text{Kilogram}^2}$$

hubble[Ωm0_,Ωd0_,Ωk0_,z_]: Hubble functions in LCDM.
H[z]: an example of hubble function with given parameters.

```
hubble[Ωm0_?NumberQ, Ωd0_?NumberQ, Ωk0_?NumberQ, z_] =
  H0 Sqrt[Ωm0 (1 + z)^3 + Ωd0 + Ωk0 (1 + z)^2];
H[z_] = hubble[Ωm0w, Ωd0w, 0, z];
```

q[Ωm0_,Ωd0_,Ωk0_,z_]:Deceleration parameter. DEF: $q[\Omega m0\_,\ \Omega d0\_,\ \Omega k0\_,\ z\_] = \frac{-1}{H[z]}\ \frac{D[D[a[t],t],t]}{D[a[t],t]}$

```
q[Ωm0_, Ωd0_, Ωk0_, z_] =
  -1 + (1 + z)/hubble[Ωm0, Ωd0, Ωk0, z] D[hubble[Ωm0, Ωd0, Ωk0, z], z];
```

(General) Friedmann equations are

$$\frac{3\,(a[t]' + k)}{a[t]^2} = 8\,\pi\,G\,\rho$$

$$2\,\frac{a[t]''}{a[t]} + \frac{a[t]' + k}{a[t]^2} = -8\,\pi\,G\,p$$

Since “zt” will be used as a parameter in most cases, we define “ztr” as a name for transition redshift, “ztrr” for transition redshift with a parameter r.
ztr[Ωm0_,Ωd0_,z_]: Transition redshift in LCDM with Ωk0 not zero with full parameters
ztrr[r_,z_]: with $r = \frac{\Omega m0}{\Omega d0}$

```
ztr[Ωm0_, Ωd0_] = (2 Ωd0/Ωm0)^(1/3) - 1;
ztrr[r_] = (2/r)^(1/3) - 1;
```

Regard “zt” as a paramter, solve out Ωd0 and Ωk0=1-Ωm0-Ωd0

$$\Omega d0 = \frac{1}{2}\,\Omega m0\,(1 + zt)^{3}$$
$$\Omega k0 = 1 - \Omega m0 - \frac{1}{2}\,\Omega m0\,(1 + zt)^{3} \tag{1}$$

CAL

Plot SN data with error bar

Import and display “SCPUnion2_AllSNe.tex”

Read the data file of SCPUnion2_mu_vs_z.txt

```
dataunionfulltmp =
  Import["DATA\\UNION2\\SCPUnion2_AllSNe.tex", "Table"];
```

[illegible]

Length of dataunionfulltmp

```
lenfull = Length[dataunionfulltmp];
```

Delete the & simbols in dataunionfulltmp

```
dataunionfull = Table[{dataunionfulltmp[[i, 1]],
     dataunionfulltmp[[i, 3]], dataunionfulltmp[[i, 5]],
     dataunionfulltmp[[i, 7]], dataunionfulltmp[[i, 9]],
     dataunionfulltmp[[i, 11]], dataunionfulltmp[[i, 13]],
     dataunionfulltmp[[i, 15]]}, {i, 1, lenfull}];
dataunionfulltable = Grid[Join[{{Name, Redshift, Mag, Stetch,
       Color, Distance, Sample, CutsFails}}, dataunionfull],
    Alignment → Left, Spacings → {2, 1}, Frame → All,
    ItemStyle → "Text", Background → {{Gray, None}, {LightGray, None}}];
```

[illegible]

[illegible]

Import and display “SCPUnion2_mu _vs _z _adjust.txt”

Import dataunion adjusted data. This data includes name, redshift, distance, error.

```
dataunion =
  Import["DATA\\UNION2\\SCPUnion2_mu_vs_z_adjust.txt", "Table"];
dataunionTable = Grid[Join[{{Name, Redshift, Distance, Error}},
     dataunion], Alignment → Left, Spacings → {2, 1}, Frame → All,
    ItemStyle → "Text", Background → {{Gray, None}, {LightGray, None}}];
```

Show dataunion table.

```
datauniontable;
```

Length of this dataunion table

```
len = Length[dataunion];
```

Simplify dataunion table and call it du

```
du = Table[{dataunion[[i, 2]],
    dataunion[[i, 3]], dataunion[[i, 4]]}, {i, 1, len}];
dutable = Grid[Join[{{Redshift, Distance, Error}}, du],
   Alignment → Left, Spacings → {2, 1}, Frame → All,
   ItemStyle → "Text", Background → {{Gray, None}, {None, None}}];
```

Show du table

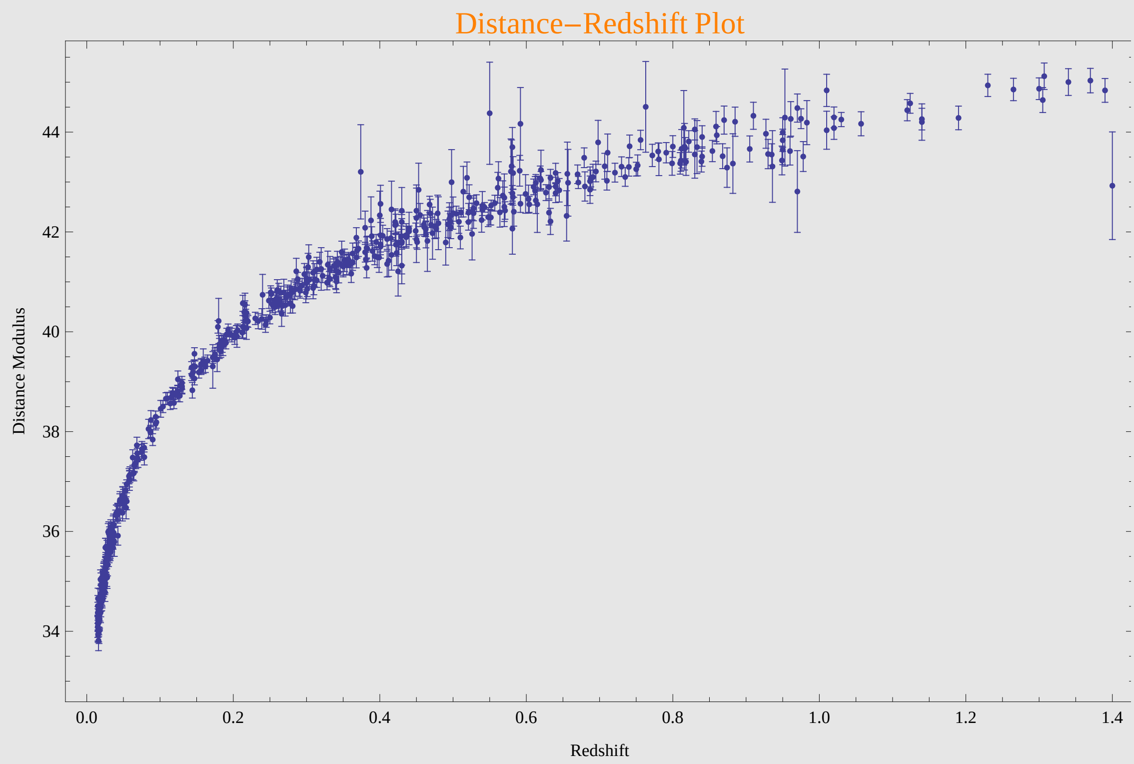
```
dutable;
```

Visualize data.

Plot du table. That is SN

```
plunion = ErrorListPlot[du, PlotRange → All];
```

```
Show[plunion, Frame -> True,
 FrameLabel -> {"Redshift", "Distance Modulus"},
 PlotLabel → Style["Distance-Redshift Plot", 16, Orange]]
```



Transition redshift, deceleration parameter, theoretically.

Theoretical Function and Values of distance/mag at different redshift.

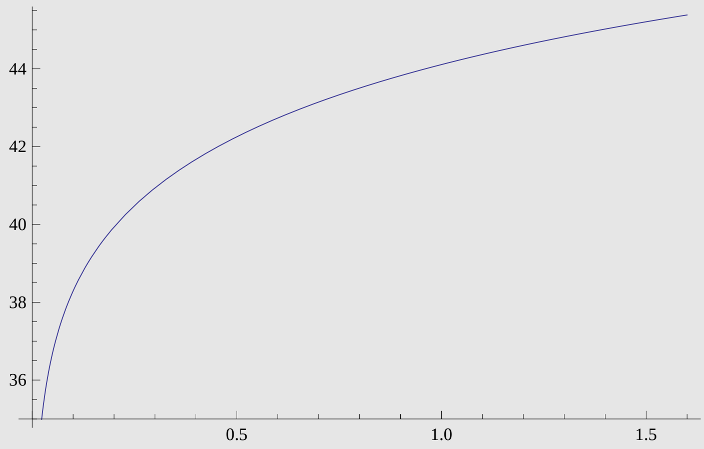
Distance at redshift z

```
dz[Ωm0_?NumberQ, Ωd0_?NumberQ, Ωk0_?NumberQ, z_?NumberQ] :=
  (1 + z) NIntegrate[1/hubble[Ωm0, Ωd0, Ωk0, tmp], {tmp, 0, z}];
```

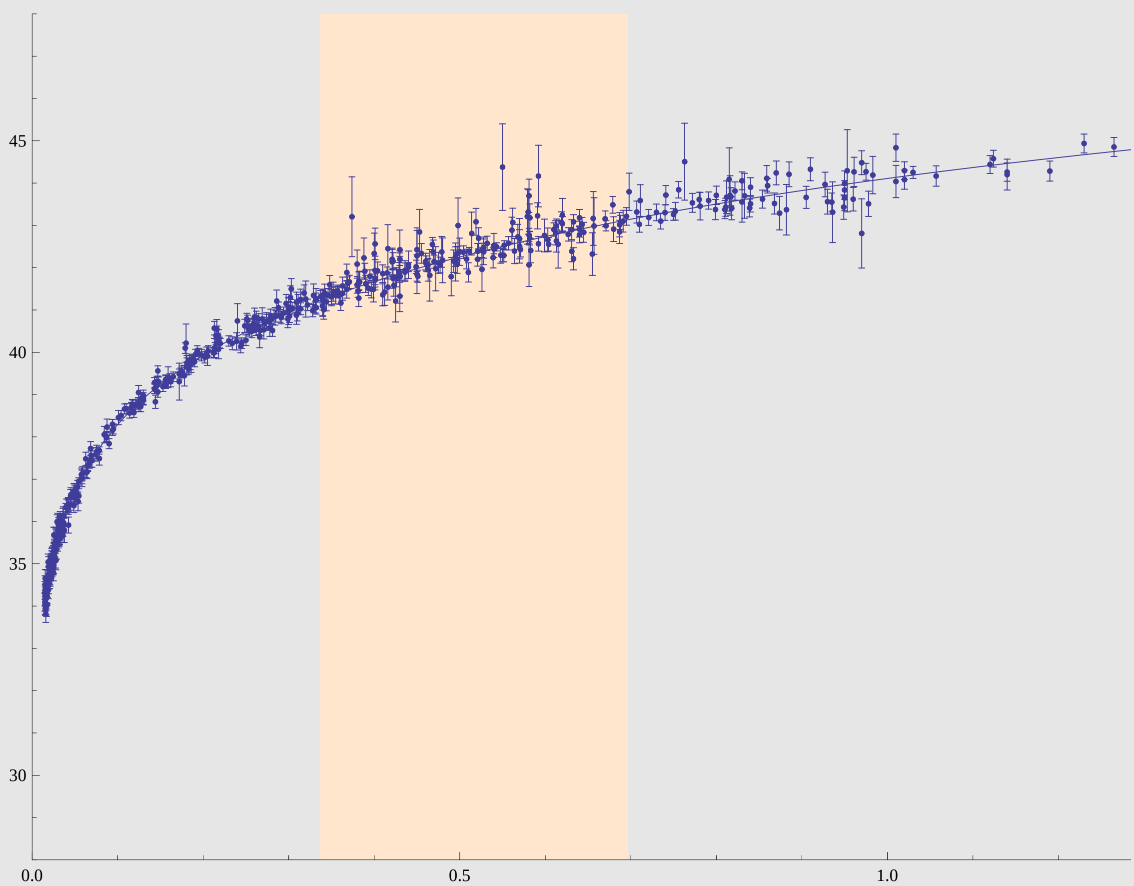
Magnitude calculated from distance. Theoretically of cause.

```
mag[Ωm0_?NumberQ, Ωd0_?NumberQ, Ωk0_?NumberQ, z_?NumberQ] :=
  5 Log[10, c * dz[Ωm0, Ωd0, Ωk0, z]] + 25;
```

```
plmagth = Plot[mag[Ωm0w, Ωd0w, 0, z], {z, 0, 1.6}]
```




```
Show[{plunion, plmagth},
 PlotRange → {{0, 1.6}, {28, 48}}, AxesOrigin → {0, 28}, Prolog →
  {LightOrange, Rectangle[{0.426 - 0.089, 28}, {0.426 + 0.27, 48}]}]
```

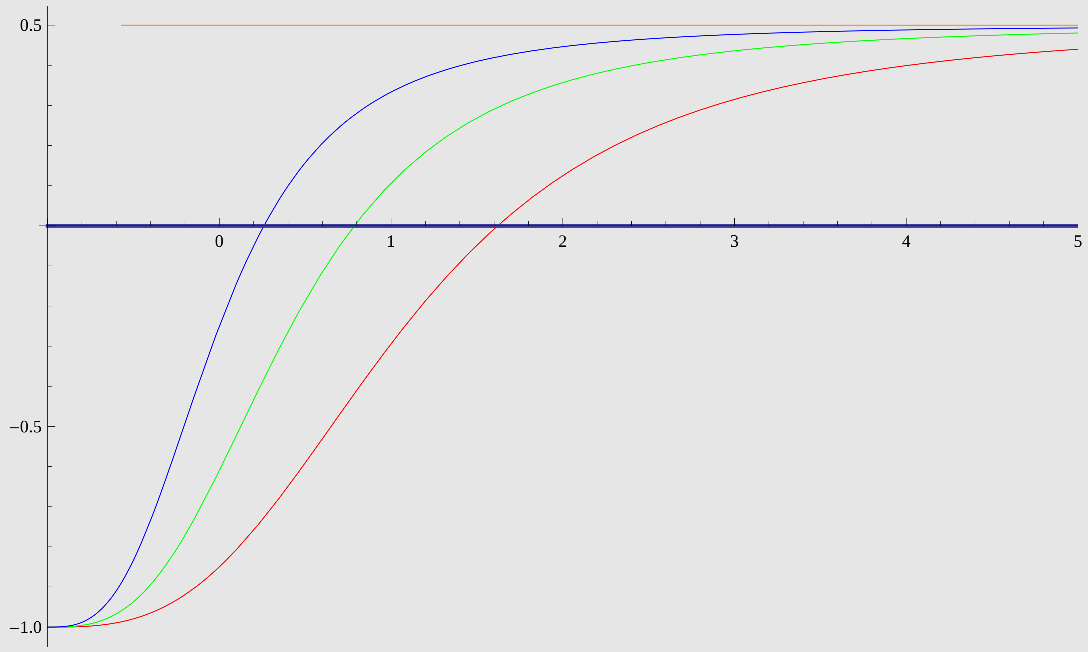


Deceleration parameter

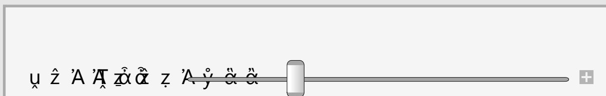
Deceleration parameter can be ploted with respect to redshift z.
Using flat FRW model. That is Ωk0=0.

```
pldec[Ωm0v_, color_] := Plot[q[Ωm0v, 1 - Ωm0v, 0, z],
  {z, -1, 5}, PlotRange → {{-1.05, 5}, {-1.05, 0.55}},
  PlotStyle → color, AxesOrigin → {-1, 0}]
```

```
Show[{pldec[0.1, Red], pldec[0.26, Green], pldec[0.5, Blue],
  pldec[1, Orange], Plot[0, {z, -1, 5}, PlotStyle → Thick]}]
```



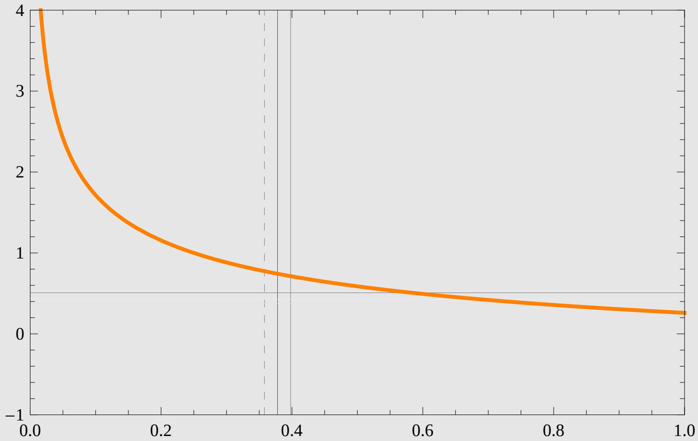

```
Manipulate[pldec[Ωm0v, {Orange, Thick}],
 {{Ωm0v, 0.26, "Matter Fraction"}, 0, 1}]
```



```
pldec[0.26, {RGBColor[1, 0.5, 0], Thickness[Large]}]
```

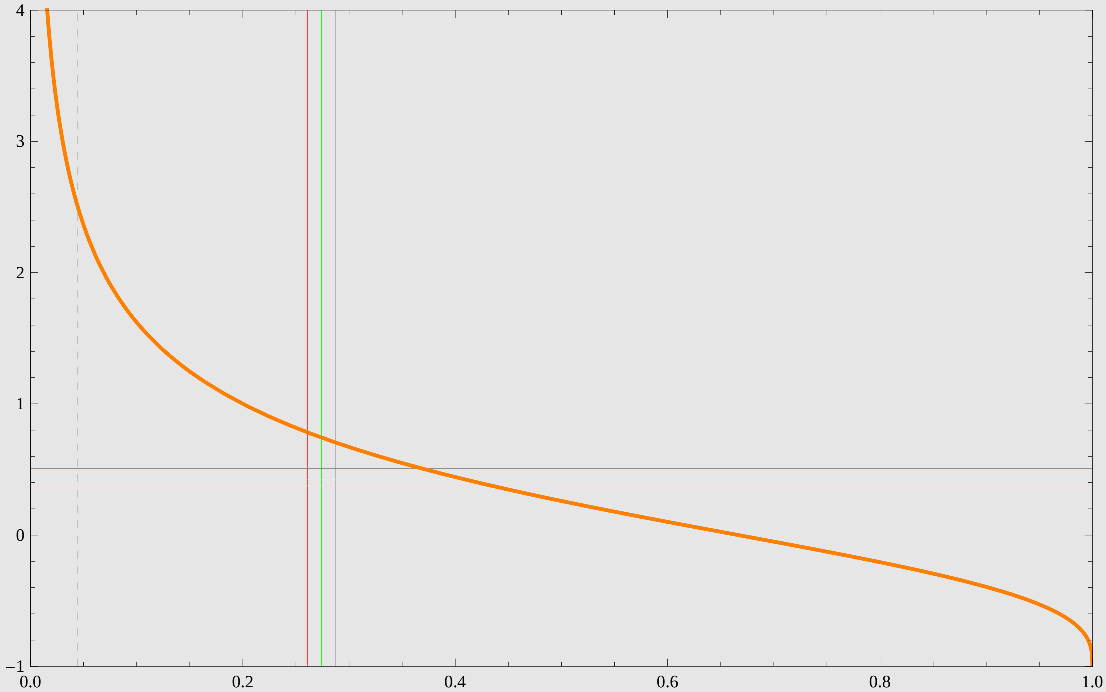
Plot deceleration parameter with respect to r. This is not related to Ωk0, i.e., the curvature of the spacetime won't affect this result.

```
pldecr = Plot[ztrr[r], {r, 0, 1}, PlotRange → {{0, 1}, {-1, 4}},
  PlotStyle → {Thick, Orange}, AxesOrigin → {0, -1}, Frame → True,
  GridLines → {{{0.358, Dashed}, {0.378, Directive[Red]}, 0.398},
    {{0.376, LightRed}, {0.426, LightBlue}, {0.508, Gray}}}]
```



In a flat universe, Ωk0=0. Then Ωd0=1-Ωm0.

```
pldecr = Plot[ztr[Ωm0, 1 - Ωm0], {Ωm0, 0, 1},
  PlotRange → {{0, 1}, {-1, 4}}, PlotStyle → {Thick, Orange},
  AxesOrigin → {0, -1}, Frame → True, GridLines →
   {{{0.044, Dashed}, {0.261, Red}, {0.274, Green}, {0.287, Gray}},
    {{0.376, LightRed}, {0.426, LightBlue}, {0.508, Gray}}}]
```



chi square fitting. Using mathematica's functions.

Definations etc

chi square defination with error only.

```
chi2unionsl[Ωm0_?NumberQ, zt_?NumberQ] :=
 ParallelSum[(du[[i, 2]] - mag[Ωm0, 1/2 Ωm0 (1 + zt)^3,
      1 - Ωm0 - 1/2 Ωm0 (1 + zt)^3, du[[i, 1]]])^2 / du[[i, 3]]^2, {i, 1, len}]
```

Define a vector/table of "theory-observation".

```
thobtab[Ωm0_, zt_] = ParallelTable[du[[i, 2]] - mag[Ωm0,
     1/2 Ωm0 (1 + zt)^3, 1 - Ωm0 - 1/2 Ωm0 (1 + zt)^3, du[[i, 1]]], {i, 1, len}];
```

Import covariance matrix data. "covm" is data with systematic and "covmns" is data without sys.

```
covm = Import["DATA/UNION2/SCPUnion2_covmat_sys.txt", "Table"];
icovm = Inverse[covm];
```

```
covmns = Import["DATA/UNION2/SCPUnion2_covmat_nosys.txt", "Table"];
icovmns = Inverse[covmns];
```

Testing the imported file.

```
du[[1]][[3]]^-2
icovmns[[1, 1]]
```

```
19.5538
```

```
19.5535
```

Define chi2 with and without systematic.

```
chi2union[Ωm0_?NumberQ, zt_?NumberQ] :=
 thobtab[Ωm0, zt].icovm.thobtab[Ωm0, zt]
```

```
chi2unionns[Ωm0_?NumberQ, zt_?NumberQ] :=
 thobtab[Ωm0, zt].icovmns.thobtab[Ωm0, zt]
```

Test the calcuation speed.

```
chi2union[0.27, 0.5] // Timing
```

```
{8.236, 560.12}
```

```
chi2unionns[0.27, 0.5] // Timing
```

```
{8.128, 735.117}
```

```
chi2unionsl[0.27, 0.5] // Timing
```

[illegible]

[illegible]

[illegible]

[illegible]

```
{0.156, 735.121}
```

Find the minimum chi2 for chi2 def. without sys.. AND Export the results to folder DATA/2/

With Systematic in data. Find the minimum of chi2.

```
Off[FindMinimum::lstol]
chi2mini = FindMinimum[chi2union[x, y], {x, 0.15}, {y, 0.4}]
On[FindMinimum::lstol]
```

```
{530.353, {x → 0.601866, y → 0.524569}}
```

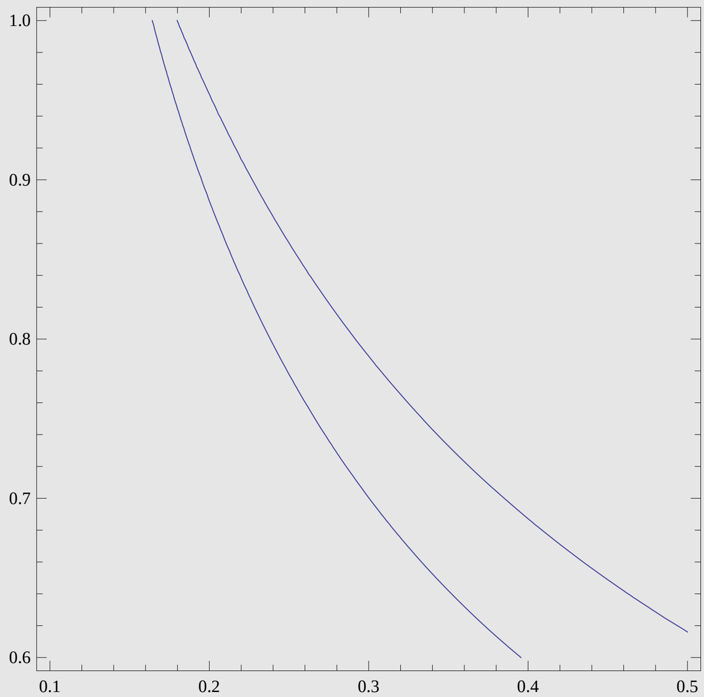
```
Export["OUTPUT\\2\\chi2union_minimum.dat", chi2mini]
```

```
OUTPUT\2\chi2union_minimum.dat
```

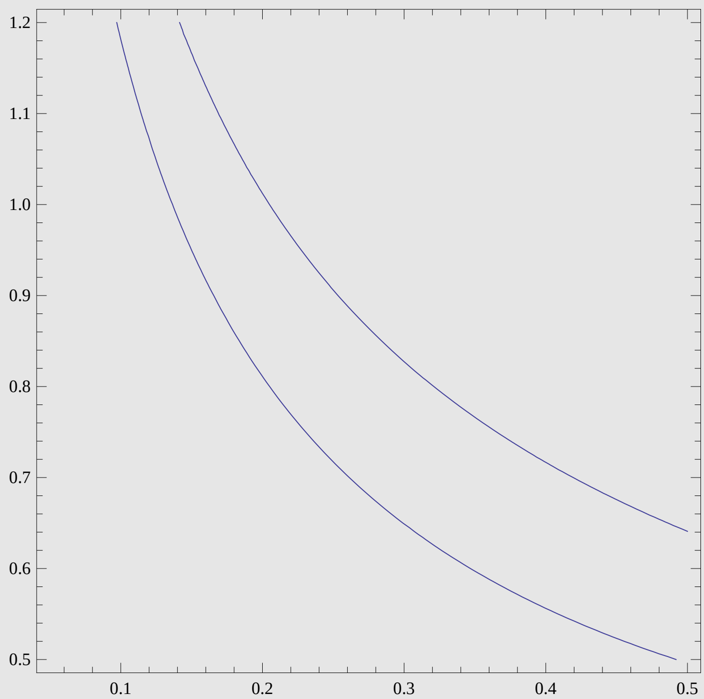
Plot the contour with confidential level 1σ and 2σ. This is a 2 parameter model. $\Delta\chi^2(1\,\sigma) = 2.30$ && $\Delta\chi^2(2\,\sigma) = 6.18$ && $\Delta\chi^2(3\,\sigma) = 11.83$.

With sys.

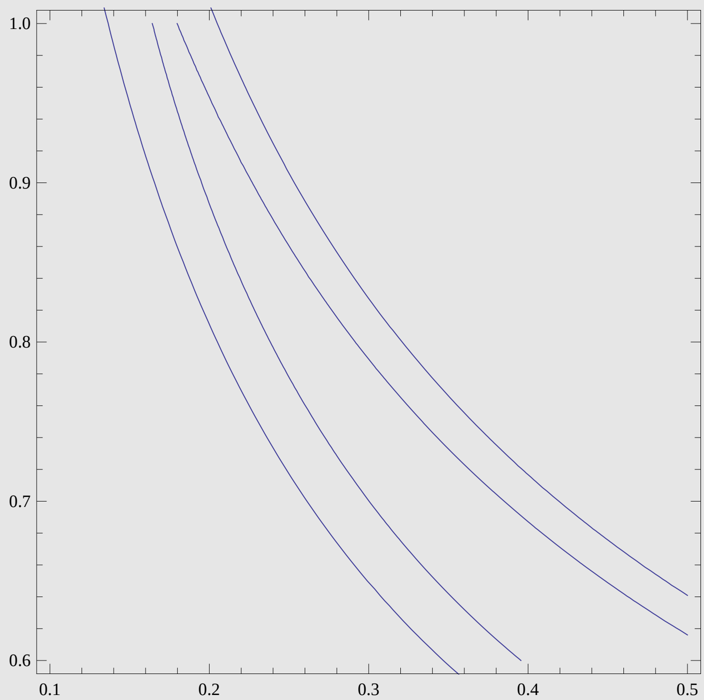
```
Off[NIntegrate::nlim]
Off[NIntegrate::slwcon]
conchi21s = ContourPlot[{chi2union[Ωm0, zt] == chi2mini[[1]] + 2.30},
  {Ωm0, 0.1, 0.5}, {zt, 0.6, 1.0}]
On[NIntegrate::nlim]
On[NIntegrate::slwcon]
```



```
Off[NIntegrate::nlim]
Off[NIntegrate::slwcon]
conchi22s = ContourPlot[{chi2union[Ωm0, zt] == chi2mini[[1]] + 6.18},
  {Ωm0, 0.05, 0.5}, {zt, 0.5, 1.2}]
On[NIntegrate::nlim]
On[NIntegrate::slwcon]
```




```
conchi2 = Show[conchi21s, conchi22s]
```



```
Export["OUTPUT\\2\\conchi21s.eps", conchi21s]
Export["OUTPUT\\2\\conchi22s.eps", conchi22s]
Export["OUTPUT\\2\\conchi2.eps", conchi2]
```

```
OUTPUT\2\conchi21s.eps
```

```
OUTPUT\2\conchi22s.eps
```

```
OUTPUT\2\conchi2.eps
```

- **With only error, not the covariance matrix.**

- **Perform a grid method**