Branches of non-symmetric critical points and symmetry breaking in nonlinear elliptic partial differential equations

—

Numerical schemes (Freefem++ and Mathematica)

Jean Dolbeault, Maria J. Esteban

March 26, 2013

Addresses. Jean Dolbeault, Maria J. Esteban: Ceremade (UMR CNRS no. 7534), Univ. Paris-Dauphine, Pl. de Lattre de Tassigny, 75775 Paris Cedex 16, France

E-mail addresses. dolbeaul@ceremade.dauphine.fr, esteban@ceremade.dauphine.fr

Contents

1 Computation of the non-symmetric branches: Freefem++

```
// This program intends to calculate the branch of minima (or at least of local minima!) of
the energy functional J for a given numbers of values of Lambda. It is designed to start
descending at a given Lambda0, and then by an iterative process construct a branch. At every
steps the starting point will be the result of the calculation for the preceding Lambda.

//-----------------------------------------------------------------------------------------
// INITIALIZATION, declaration of variables, sizes, numbers of iterations and steps, etc
//-----------------------------------------------------------------------------------------
real  Lambda,  d, p, q, eps, rho, l, m, e, mu, nu, kappa, bstar ;
int L=12, N0= 4,  N = L*N0, M= pi*N0, i ;

int nsteps = 80;  // number of steps in the loop for constructing the branch
int K=250; // number of iterations pour chaque point de la branche
real sign = +1; // +1 pour monter en Lambda, -1 pour aller vers le bas dans la branche,
diminumant le Lambda.
real hhh = 0.1;  // largeur du pas pour l'iteration quand on construit la branche. 0.1 est
grossier, 0.01 est fin.
int ww=1;
cout.precision(16) ;

// values of theta (def of energy), dimension, power p (in the energy)
real theta=1;
d=5 ;
p=3.15 ;
eps=1e-1;
real ff=1.;
q=p/(p-2);
Lambda=2.85;
ofstream file("evolutionofenergies-methode-iterative-p=315b",append); // nom fichier sortie

// definition of functions and values to calculate the radially symetric function ustar
func LambdaFS=4*(d-1)/(p^2-4) ;
func BFS =   ((p-2)*sqrt(Lambda))/2   ;
bstar =BFS;

// definition of fuction allowing to compute normalization constants in the integrals, to
recover the variables that are not included in the definition of the integrals, and which
should be there to calculare integrals over R^d
func FSd = (2* (pi)^(d/2))/(tgamma(d/2));
real Sd=FSd;

func FHnouv= (sqrt(pi)* tgamma( (-1 + d)/2))/ tgamma(d/2);
real Hnouv=FHnouv;

func FCoeffd=  (Sd/Hnouv)^((p-2)/p);
real Coeffd=FCoeffd;

func FCoeffdlin=Sd/Hnouv;
real Coeffdlin=FCoeffdlin;

// definition of domain. We compute in a rectangle. The boundary conditions will be defined
later
border b1(t=0,1){ real tt=t;x=L*(   tt        ); y=0; label=1;}
border b2(t=0,1){ x=L; y=pi*t; label=2;}
border b3(t=1,0){ real tt=t;x=L*(    tt    ); y=pi; label=3;}
border b4(t=1,0){ x=0; y=pi*t; label=4;}
mesh Th = buildmesh(b1(N)+b2(M)+b3(N)+b4(M));
real ccc=0.5;//  ccc= 0 +> regulier , ccc= 1 gros raffinement en x=0.
func hx = 1./N0*(1+ccc*x)/(1+ccc*L);
func hy = 1./N0;
```

```
Th= adaptmesh(Th,1/(hx*hx),0,1./(hy*hy),IsMetric=1,nbvx=1000000);
Th= adaptmesh(Th,1/(hx*hx),0,1./(hy*hy),IsMetric=1,nbvx=1000000);
Th= adaptmesh(Th,1/(hx*hx),0,1./(hy*hy),IsMetric=1,nbvx=1000000);
savemesh(Th,"Th.msh");

plot(Th,wait=0);
fespace Vh(Th,P2);
func ustar= (cosh(bstar*x))^(-2/(p-2)) ;  // FAUT PARTIR AVEC GRADIENT NEGATIF
func uconstante=1 ;
Vh us= abs(ustar);
Vh u=us;

//-------------------------------------------------------------------------------------
// Definition of energy functional, its derivative, and intermediate integrals and norms
func real J(real[int] & udof)  // % udof??
{
     Vh u;
     u[]=udof;
     real A= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)+Lambda*u*u) * (sin(y))^(d-2))) ;
     real C=  int2d(Th) ( (abs(u)^p)* (sin(y))^(d-2) );
     real energy= Coeffd* 2*A*((2*C)^(-2./p)) ; // % J(u) ?   %; ?? pour avoir la valeur de
J(u*)
     return energy;
}
//---------------------------------

real Jus = J(us[]);
kappa=Jus; // kappa is going to be used in the iteration
// cout << "AAA = " << A <<, " BBB =" << B << endl;  // this command prints AAA and BBB in
the computing (terminal) window
// exit(0); //  this commnad allows to stop the calculation where to wish. If you have a
"cout" line before it, you can check what the program is doing by checking AAA and BBB just
before the programs stops

func real A(real[int] & udof)  // % udof??
{
     Vh u;
     u[]=udof;
     real A= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)+Lambda*u*u) * (sin(y))^(d-2))) ;
     real kineticenergy= Coeffdlin* A ; // % J(u) ?   %; ?? pour avoir la valeur de J(u*)
     return kineticenergy;
}
//---------------------------------

real Aus = A(us[]);
func real C(real[int] & udof)
{
     Vh u;
     u[]=udof;
          real C=  int2d(Th) ( (abs(u)^p)* (sin(y))^(d-2) );
     real Lpnorm= Coeffdlin* C ;
     return Lpnorm;
}
//---------------------------------

real Cus = C(us[]);
func real L2(real[int] & udof)
{
     Vh u;
     u[]=udof;
```

```
    real M=  int2d(Th) ( (abs(u)^2)* (sin(y))^(d-2) );
    real Mass= Coeffdlin* M ;
    return Mass;
}
//---------------------------------

real L2us = L2(us[]);   // all these integrals are in fact half of the norms in the whole
space, except J for which the 2 there is already taken into account.
func real D(real[int] & udof)
{
    Vh u;
    u[]=udof;
    real B= int2d(Th) ( (dx(u)*dx(u)+dy(u)*dy(u)) * (sin(y))^(d-2)) ;
    real Direnergy= Coeffdlin* B ;
    return Direnergy;
}
//---------------------------------

real Dus = D(us[]);
/*
func real T(real[int] & udof)
{
    Vh u;
    u[]=udof;
    real A= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)+Lambda*u*u) * (sin(y))^(d-2))) ;
    real D= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)) * (sin(y))^(d-2))) ;
    real C=  int2d(Th) ( (abs(u)^p)* (sin(y))^(d-2) );
    Vh upmoins2=abs(u)^(p-2);
    real Vint = int2d(Th) (( upmoins2*u^2 * (sin(y))^(d-2))) ;
    real M= int2d(Th) ( (abs(u)^2)* (sin(y))^(d-2) );
    real vpenergy=  ((2*Coeffdlin*D)- Coeffd* 2*A*((2*C)^(-2./p)) * 2*Coeffdlin*Vint*
(2*Coeffdlin*C)^((2-p)/p)       )/(2*Coeffdlin*M) ;
    return vpenergy;
}

real  Tus = T(us[]);
func real Tkappa(real[int] & udof)
{
    Vh u;
    u[]=udof;
    real A= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)+Lambda*u*u) * (sin(y))^(d-2))) ;
    real D= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)) * (sin(y))^(d-2))) ;
    real C=  int2d(Th) ( (abs(u)^p)* (sin(y))^(d-2) );
    Vh upmoins2=abs(u)^(p-2);
    real Vint = int2d(Th) (( upmoins2*u^2 * (sin(y))^(d-2))) ;
    real M= int2d(Th) ( (abs(u)^2)* (sin(y))^(d-2) );
//   real vpenergy=  ((2*AA)-(2*A*((2*C)^(-2./p)))* Vint* ( (2*C)^(2-p)/p) )/(2*M) ; // %
J(u) ?   %; ?? pour avoir la valeur de J(u*)
real vpenergy=  ((2*Coeffdlin*D)- kappa * 2*Coeffdlin*Vint* (2*Coeffdlin*C)^((2-p)/
p)       )/(2*Coeffdlin*M) ; // %     J(u) ?   %; ?? pour avoir la valeur de J(u*)
    return vpenergy;
}

real Tkappaus = Tkappa(us[]);
func real F0(real[int] & udof)  // % udof??
{
    Vh u;
    u[]=udof;
    real A= int2d(Th) (( (dx(u)*dx(u)+dy(u)*dy(u)+Lambda*u*u) * (sin(y))^(d-2))) ;
    real C=  int2d(Th) ( (abs(u)^p)* (sin(y))^(d-2) );
```

```
    real M= int2d(Th) ( (abs(u)^2)* (sin(y))^(d-2) );
    real Fequalto0=  (   (2*A)-(kappa*   ((2*C)^(2./p)) )    )      /(2*M) ; // % J(u) ?
%; ?? pour avoir la valeur de J(u*)
    return Fequalto0;
}

real F0us=F0(us[]);
*/

func real[int]  DJ(real[int] & udof)
{
    Vh u;
    u[]=udof;
    real A=  int2d(Th) ((( dx(u)*dx(u)+dy(u)*dy(u) +Lambda*u*u )* (sin(y))^(d-2))) ;
    real C=   int2d(Th) (    (abs(u)^p)* (sin(y))^(d-2)     );
    real J= Coeffd*2*A*((2*C)^(-2./p))  ;  // % J(u) ?   %; ?? pour avoir la valeur de
J(u*)
    cout  << " JJJJ = " << J <<  endl;
    varf vDJ(notusedvalue,v) = int2d(Th)(  (  ((2*J* (sin(y))^(d-2)))*  ( (1./
(2*A))*(dx(u)*dx(v)+dy(u)*dy(v)+Lambda*u*v)-(1./(2*C))*abs(u)^(p-2)*u*v ))) +
on(2,notusedvalue=0); //   notused ?    varf ??   notusedvalue?????   Dj  ou DJ ?

    real[int] b=vDJ(0,Vh);  //    % b ?
    return b;
}
//----------------------------------

// calcul vp du Hessien -- to calculate the second eigenvalue of it, which is the first in
the non radially symmetric direction.
cout  << " calcul J et DJ fini " << kappa <<  endl;
{
real Ju = J(u[]);
//real Tu=T(u[]);
real Cu=C(u[]);
real Au=A(u[]);
real Du=D(u[]);
real L2u=L2(u[]);
real sigma=0;
varf vH(w,v) =  int2d(Th)(2*Ju*( ((sin(y))^(d-2)) * (
    (dx(w)*dx(v)+dy(w)*dy(v) + Lambda*w*v)/(2*Au)  - (  1./(2*Cu)  )    *       (p-1)*
(w*v*(abs(u)^(p-2)) )
    ))
    )+ on(2,w=0);
varf vM(w,v) =   int2d(Th)(      w*v*((sin(y))^(d-2))       ) ;

matrix H=vH(Vh,Vh,solver=UMFPACK); // matrix
matrix M=vM(Vh,Vh); // matrix
int nev = 2;
real[int] ev(nev); // to store nev eigein value
Vh[int] eV(nev);   // to store nev eigen vector

int k=EigenValue(H,M,sym=true,sigma=sigma,value=ev,vector=eV);
k = min(k,nev);
ofstream foof("VmuP-Hessien-"+Lambda+".data") ;
foof << k << endl;
for(int i=0;i<k;++i)
{
    foof << i << endl;
    foof << ev[i] << endl; // foof << eV[i][] << endl;
    plot(eV[i],cmm=" " + ev[i], wait=0 )  ;
```

```
    cout  << " iHess = " << i << " vpHessien = " << ev[i] <<   endl;
}
u=u+eps*eV[1];
}

plot(u,dim=2,fill=1,ps="u-"+Lambda+".eps",cmm="Lambda="+Lambda,value=1);
// if the second eigenvalue was negative, then the new u is near us, but has lower energy
and is not radially symmetric anymore
//---------------------------------------------------------------------------------------------
-----------------------------------


//---------------------------------
// GRADIENT CONJUGUE PRECONDITIONNE
//---------------------------------
int option = 1;
verbosity=1;
cout << " init J = " << J(u[]) << endl;
if( option==0)  // methode de descent a pas simple ...
{
 real rho = 1;
 for(int i=0;i< 1000; ++i)
 {
     real[int] w = DJ(u[]);       //   % c'est un reel sans appliquer a rien ? c'est quoi le
carre ?
     Vh ww; ww[]=w; plot(ww,wait=0);
     real err= sqrt(w.l2) ;
     cout<< " i = " << i << " err " << err << "  J = " << J(u[]) << endl;
     if(err < 1e-5) break;
     u[] -= w*rho;
  }
} else  // methode de Gradient Conjugue NL Precondionne ...
{
 varf a(u,v) =  int2d(Th) (( dx(u)*dx(v)+dy(u)*dy(v) +Lambda*u*v )* (sin(y))^(d-2)) +
on(2,u=0);  //   modifier varf ? preconditionneur a modifier?
 matrix A= a(Vh,Vh,solver=UMFPACK);
 func real[int]  PreCon(real[int] & udof)
      {
        real[int] b=A^-1*udof;
        return b;
       }
 NLCG(DJ,u[],eps=-1.e-10,verbosity=50,nbiter=100,precon=PreCon);
      // test arret sur || g^2|| < 1.e-8  si negatif sinon le test
      // || g^2||  < ||g_0^2|| eps  ou g_0 est le gradiant initial ..
 plot(u,dim=2,fill=1,ps="u-"+Lambda+".eps",cmm="Lambda="+Lambda,value=1);
 cout << " finnnnal  J = " << J(u[]) << " Difference =" << Jus-J(u[]) << endl;
 cout << " JafterGC= " << J(u[]) << endl;
//plot(u,dim=3,fill=1);
real Ju=J(u[]);
// real Tu = T(u[]);
// real Tkappau = Tkappa(u[]);
}

real Ju=J(u[]);
kappa=Ju;
//real Tu = T(u[]);
//real Tkappau = Tkappa(u[]);
//------------------------------------------
```

```
//----------------------------------------
// LOOP FOR THE CONSTRUCTION OF THE BRANCH
//----------------------------------------

Vh v=u;
for ( int i=0; i<nsteps;i++)
{
real Ju = J(u[]);
kappa=Ju+sign*hhh ;
for ( int i=0; i<K;i++)

{
 Ju = J(u[]);
 Vh uold=u;
 real Lambdaold=Lambda;
 real Jold=Ju;
 real  Cu=C(u[]);
 Vh V= (abs(u)^(p-2))* (2*Cu)^((2-p)/p) ;
 real normeqV=int2d(Th)( abs(V)^(p/(p-2))* ((sin(y))^(d-2)) );
 real sigma=-80;
 varf vH(w,v) =   int2d(Th)( (   ( (dx(w)*dx(v)+dy(w)*dy(v))  - (kappa*w*v*V) - (sigma*w*v))
* ((sin(y))^(d-2))    )) + on(2,w=0);
 varf vS(w,v) =   int2d(Th)(       w*v*((sin(y))^(d-2))       ) ;

 matrix H=vH(Vh,Vh,solver=UMFPACK); // matrix
 matrix S=vS(Vh,Vh); // matrix
 int nev = 10;
 real[int] ev(nev); // to store nev eigein value
 Vh[int] eV(nev);   // to store nev eigen vector
 int k=EigenValue(H,S,sym=true,sigma=sigma,value=ev,vector=eV);
 k = min(k,nev);
 ev.sort; // tri croissant

 plot(eV[0],cmm=" " + ev[0], wait=0 )  ;
 // plot (V,cmm=" " + ev[0], wait=ww )  ;
 //plot(eV[1],cmm=" " + ev[1], wait=ww )  ;
 u=eV[0];
 Lambda=-ev(0);
 Vh uup=abs(u)^p;
 real normapres=  int2d(Th)(   uup* ((sin(y))^(d-2)) );
 bstar =BFS;
 Vh usboucle=ustar;
 real Jusboucle = J(usboucle[]);
 Ju = J(u[]);
 real L2u = L2(u[]);
 real Du = D(u[]);
 Cu=C(u[]);
 cout << "------------ Lambda = " << Lambda << ", bstar = " << bstar << ", Jusboucle = " <<
Jusboucle << endl;

 // exit(0);
 real Diff1=int2d(Th) ( (abs(u-uold)^2)* (sin(y))^(d-2) );
 real Diff2=abs(Lambdaold-Lambda);
 real Diff3=abs(Jold-Ju);
 real Diff=Diff1+Diff2+Diff3;

 if ( Diff< 1e-10){  break; } // test d'arret
 if (Lambda<1) break;
}
```

```
//new computation of radial minimizer, with updated values of parameters
bstar =BFS;
Vh ushorsboucle=ustar;
real Jushorsboucle = J(ushorsboucle[]);
Ju = J(u[]);
real L2u = L2(u[]);
real Du = D(u[]);
real  Cu=C(u[]);
file << " ihors boucle = " << i << ", kappa = " << kappa << ",  Lambda(kappa) = " << Lambda
<< ",  J(Lambda(kappa)) = " << Ju << ", Jushorsboucle = " << Jushorsboucle << ", L2 = " <<
2*L2u << ", L2gradient = " << 2*Du <<  ", Lp = " << 2*Cu << endl;

// we recover this value in the file defined at the beginning of the program. The norms are
good now, since we have multiplied the computed integrals by 2. Only J is not multiplied by
2 because its definition took that into account.
Th=adaptmesh(Th,u,err=0.001, nbvx=10000); // 0.001 = error level
plot(Th,wait=0);
u=u;
}
// END OF THE LOOP
//----------------
```

2 Computation of the non-symmetric branches: Mathematica

Branches of non symmetric optimal functions

Data for the computation of the branches

Numerical values below have been produced with a Frefem++ code using self-adaptive grids.

Values for p = 2.8, d = 5, adaptative mesh - 8 November 2011
i, kappa, Lambda(kappa), J(Lambda(kappa)), L2, L2gradient, Lp
Values of Lambda < 5

```
L811a = {{0, 19.7231, 5.94199, 19.7228, 21.2178, 39.4784, 213.009, 79.1182},
    {1, 19.6221, 5.88597, 19.6221, 21.0463, 39.4784, 207.997, 77.8922},
    {2, 19.5221, 5.8295, 19.5221, 20.8732, 39.4784, 202.231, 76.4645},
    {3, 19.4221, 5.77377, 19.4221, 20.702, 39.4784, 196.534, 75.0541},
    {0, 19.3717, 5.74463, 19.3715, 20.6123, 39.4784, 192.986, 74.164},
    {1, 19.271, 5.69093, 19.271, 20.447, 39.4784, 188.052, 72.9547},
    {2, 19.171, 5.63704, 19.171, 20.2811, 39.4784, 182.526, 71.5875},
    {3, 19.071, 5.58389, 19.071, 20.117, 39.4784, 177.07, 70.2376},
    {4, 18.971, 5.53145, 18.971, 19.955, 39.4784, 171.68, 68.9044},
    {5, 18.871, 5.47972, 18.871, 19.795, 39.4784, 166.357, 67.5882},
    {6, 18.771, 5.42871, 18.771, 19.6369, 39.4784, 161.099, 66.2882},
    {7, 18.671, 5.3784, 18.671, 19.4808, 39.4784, 155.907, 65.0048},
    {8, 18.571, 5.32879, 18.571, 19.3267, 39.4784, 150.78, 63.7381},
    {9, 18.471, 5.27988, 18.471, 19.1745, 39.4784, 145.718, 62.4874},
    {10, 18.371, 5.23166, 18.371, 19.0243, 39.4784, 140.718, 61.2525},
    {11, 18.271, 5.18412, 18.271, 18.876, 39.4784, 135.783, 60.034},
    {12, 18.171, 5.13726, 18.171, 18.7297, 39.4784, 130.91, 58.8311},
    {13, 18.071, 5.09108, 18.071, 18.5853, 39.4784, 126.1, 57.6441},
    {14, 17.971, 5.04557, 17.971, 18.4428, 39.4784, 121.351, 56.4726},
    {15, 17.871, 5.00072, 17.871, 18.3022, 39.4784, 116.664, 55.3168},
    {16, 17.771, 4.95653, 17.771, 18.1635, 39.4784, 112.038, 54.1763},
    {17, 17.671, 4.913, 17.671, 18.0266, 39.4784, 107.471, 53.051},
    {18, 17.571, 4.87012, 17.571, 17.8917, 39.4784, 102.964, 51.9407},
    {19, 17.471, 4.82788, 17.471, 17.7586, 39.4784, 98.5174, 50.8456},
    {20, 17.371, 4.78628, 17.371, 17.6273, 39.4784, 94.1285, 49.7652},
    {21, 17.271, 4.74532, 17.271, 17.498, 39.4784, 89.7988, 48.6998},
    {22, 17.171, 4.70499, 17.171, 17.3704, 39.4784, 85.5258, 47.6488},
    {23, 17.071, 4.66529, 17.071, 17.2447, 39.4784, 81.3105, 46.6124},
    {24, 16.971, 4.6262, 16.971, 17.1208, 39.4784, 77.1528, 45.5907},
    {25, 16.871, 4.58774, 16.871, 16.9987, 39.4784, 73.0506, 44.5831},
    {26, 16.771, 4.54988, 16.771, 16.8784, 39.4784, 69.0046, 43.5897},
    {27, 16.671, 4.51263, 16.671, 16.7599, 39.4784, 65.014, 42.6105},
    {28, 16.571, 4.47599, 16.571, 16.6431, 39.4784, 61.0784, 41.6452},
    {29, 16.471, 4.43994, 16.471, 16.5282, 39.4784, 57.1974, 40.6939},
    {30, 16.371, 4.40448, 16.371, 16.415, 39.4784, 53.3705, 39.7564},
    {31, 16.271, 4.36961, 16.271, 16.3035, 39.4784, 49.5977, 38.8326},
    {32, 16.171, 4.33533, 16.171, 16.1938, 39.4784, 45.8845, 37.924},
    {33, 16.071, 4.30162, 16.071, 16.0858, 39.4784, 42.2355, 37.0316},
    {34, 15.971, 4.26849, 15.971, 15.9796, 39.4784, 38.6682, 36.1598},
    {35, 15.871, 4.23593, 15.871, 15.875, 39.4784, 35.2331, 35.3208},
    {36, 15.771, 4.20392, 15.771, 15.7722, 39.4784, 32.0594, 34.5462},
    {37, 15.671, 4.17244, 15.671, 15.6709, 39.4784, 29.4334, 33.9056},
    {38, 15.571, 4.14136, 15.571, 15.5708, 39.4784, 27.7721, 33.5005},
```

```
{39, 15.471, 4.11041, 15.471, 15.4709, 39.4784, 27.1022, 33.3368},
{40, 15.371, 4.07944, 15.371, 15.371, 39.4784, 26.8411, 33.2726},
{41, 15.2709, 4.0485, 15.2709, 15.2709, 39.4784, 26.634, 33.2214},
{42, 15.1698, 4.0176, 15.1698, 15.1699, 39.4784, 26.4313, 33.174},
{43, 15.0705, 3.98673, 15.0705, 15.0705, 39.4784, 26.2246, 33.12},
{44, 14.9708, 3.95589, 14.9708, 14.9708, 39.4784, 26.0289, 33.0695},
{45, 14.8695, 3.92508, 14.8695, 14.8695, 39.4784, 25.8339, 33.0239},
{46, 14.7699, 3.89433, 14.7699, 14.77, 39.4784, 25.6239, 32.9689},
{47, 14.6698, 3.86359, 14.6698, 14.6698, 39.4784, 25.4209, 32.917},
{48, 14.5708, 3.8329, 14.5708, 14.5708, 39.4784, 25.2131, 32.86},
{49, 14.4709, 3.80223, 14.4709, 14.4709, 39.4784, 25.02, 32.8094},
{50, 14.3707, 3.77161, 14.3707, 14.3707, 39.4784, 24.811, 32.7551},
{51, 14.2711, 3.74101, 14.2711, 14.2711, 39.4784, 24.614, 32.7016},
{52, 14.1704, 3.71045, 14.1704, 14.1704, 39.4784, 24.4155, 32.6512},
{53, 14.0701, 3.67993, 14.0701, 14.0701, 39.4784, 24.2111, 32.5974},
{54, 13.9711, 3.64945, 13.9711, 13.9711, 39.4784, 24.0122, 32.5405},
{55, 13.8702, 3.619, 13.8702, 13.8702, 39.4784, 23.8086, 32.4882},
{56, 13.7706, 3.5886, 13.7706, 13.7707, 39.4784, 23.6174, 32.4346},
{57, 13.6711, 3.55823, 13.6711, 13.6711, 39.4784, 23.4069, 32.3752},
{58, 13.5712, 3.52789, 13.5712, 13.5713, 39.4784, 23.2128, 32.3209},
{59, 13.4711, 3.49758, 13.4711, 13.4712, 39.4784, 23.011, 32.2649},
{60, 13.3711, 3.46732, 13.3711, 13.3711, 39.4784, 22.8139, 32.2098},
{61, 13.2697, 3.4371, 13.2697, 13.2697, 39.4784, 22.6219, 32.1602},
{62, 13.171, 3.40692, 13.171, 13.171, 39.4784, 22.4093, 32.0952},
{63, 13.0716, 3.37677, 13.0716, 13.0716, 39.4784, 22.2171, 32.0381},
{64, 12.9706, 3.34665, 12.9706, 12.9706, 39.4784, 22.0209, 31.9847},
{65, 12.8712, 3.31658, 12.8712, 12.8712, 39.4784, 21.8184, 31.9237},
{66, 12.771, 3.28654, 12.771, 12.771, 39.4784, 21.6266, 31.8681},
{67, 12.6702, 3.25655, 12.6702, 12.6702, 39.4784, 21.4272, 31.8121},
{68, 12.571, 3.2266, 12.571, 12.571, 39.4784, 21.2367, 31.7526},
{69, 12.471, 3.19669, 12.471, 12.471, 39.4784, 21.0226, 31.6887},
{70, 12.3717, 3.16681, 12.3717, 12.3717, 39.4784, 20.8368, 31.6299},
{71, 12.2701, 3.13697, 12.2701, 12.2702, 39.4784, 20.6385, 31.5752},
{72, 12.1716, 3.10718, 12.1716, 12.1716, 39.4784, 20.4416, 31.5098},
{73, 12.0713, 3.07743, 12.0713, 12.0713, 39.4784, 20.249, 31.4513},
{74, 11.9712, 3.04772, 11.9712, 11.9712, 39.4784, 20.0524, 31.3906},
{75, 11.8711, 3.01804, 11.8711, 11.8711, 39.4784, 19.8575, 31.3296},
{76, 11.7716, 2.98841, 11.7716, 11.7716, 39.4784, 19.664, 31.2666},
{77, 11.671, 2.95883, 11.671, 11.6711, 39.4784, 19.4643, 31.2051},
{78, 11.5718, 2.92928, 11.5718, 11.5718, 39.4784, 19.273, 31.1406},
{79, 11.4712, 2.89977, 11.4712, 11.4712, 39.4784, 19.0799, 31.0805},
{80, 11.3708, 2.87031, 11.3708, 11.3708, 39.4784, 18.8919, 31.0207},
{81, 11.2713, 2.8409, 11.2713, 11.2713, 39.4784, 18.6854, 30.9509},
{82, 11.1721, 2.81152, 11.1721, 11.1721, 39.4784, 18.4979, 30.8856},
{83, 11.071, 2.78219, 11.071, 11.071, 39.4784, 18.3055, 30.8254},
{84, 10.9714, 2.75291, 10.9714, 10.9714, 39.4784, 18.1108, 30.7581},
{85, 10.8722, 2.72367, 10.8722, 10.8722, 39.4784, 17.9192, 30.6897},
{86, 10.7708, 2.69447, 10.7708, 10.7709, 39.4784, 17.7323, 30.6308},
{87, 10.6712, 2.66532, 10.6712, 10.6712, 39.4784, 17.5294, 30.5592},
{88, 10.5722, 2.63621, 10.5722, 10.5722, 39.4784, 17.3437, 30.4903},
{89, 10.4707, 2.60714, 10.4707, 10.4707, 39.4784, 17.154, 30.4295},
{90, 10.3718, 2.57813, 10.3718, 10.3718, 39.4784, 16.9558, 30.3545},
{91, 10.2722, 2.54916, 10.2722, 10.2722, 39.4784, 16.7712, 30.2863},
{92, 10.1714, 2.52024, 10.1714, 10.1715, 39.4784, 16.5797, 30.2202},
{93, 10.0715, 2.49136, 10.0715, 10.0715, 39.4784, 16.394, 30.1521},
```

{94, 9.972, 2.46253, 9.972, 9.97201, 39.4784, 16.198, 30.0776},
{95, 9.87202, 2.43375, 9.87202, 9.87203, 39.4784, 16.0149, 30.0093},
{96, 9.77159, 2.40502, 9.77159, 9.77161, 39.4784, 15.8208, 29.9381},
{97, 9.67193, 2.37634, 9.67194, 9.67195, 39.4784, 15.6362, 29.8666},
{98, 9.5723, 2.34771, 9.5723, 9.5723, 39.4784, 15.4456, 29.7919},
{99, 9.47129, 2.31913, 9.47129, 9.47131, 39.4784, 15.2589, 29.7241},
{100, 9.3718, 2.29059, 9.37181, 9.37182, 39.4784, 15.072, 29.6488},
{101, 9.27222, 2.26211, 9.27222, 9.27223, 39.4784, 14.8798, 29.571},
{102, 9.17157, 2.23368, 9.17157, 9.17158, 39.4784, 14.7053, 29.5043},
{103, 9.07214, 2.2053, 9.07214, 9.07215, 39.4784, 14.5041, 29.4207},
{104, 8.97172, 2.17697, 8.97172, 8.97174, 39.4784, 14.3315, 29.3523},
{105, 8.87223, 2.1487, 8.87223, 8.87224, 39.4784, 14.1329, 29.2683},
{106, 8.77205, 2.12047, 8.77205, 8.77206, 39.4784, 13.9542, 29.1949},
{107, 8.67168, 2.0923, 8.67168, 8.67169, 39.4784, 13.7674, 29.1181},
{108, 8.57219, 2.06419, 8.57219, 8.5722, 39.4784, 13.5762, 29.0345},
{109, 8.47219, 2.03613, 8.47219, 8.4722, 39.4784, 13.3992, 28.9584},
{110, 8.37159, 2.00812, 8.3716, 8.37162, 39.4784, 13.2142, 28.8811},
{111, 8.27242, 1.98018, 8.27242, 8.27243, 39.4784, 13.0235, 28.7933},
{112, 8.17176, 1.95228, 8.17176, 8.17178, 39.4784, 12.8493, 28.7192},
{113, 8.07195, 1.92445, 8.07195, 8.07196, 39.4784, 12.6626, 28.6345},
{114, 7.97236, 1.89667, 7.97236, 7.97237, 39.4784, 12.4733, 28.5464},
{115, 7.87194, 1.86894, 7.87194, 7.87195, 39.4784, 12.3027, 28.4701},
{116, 7.77152, 1.84128, 7.77152, 7.77154, 39.4784, 12.1169, 28.3858},
{117, 7.67242, 1.81368, 7.67242, 7.67242, 39.4784, 11.9315, 28.294},
{118, 7.57198, 1.78614, 7.57198, 7.572, 39.4784, 11.7473, 28.2085},
{119, 7.47267, 1.75865, 7.47267, 7.47267, 39.4784, 11.5712, 28.1199},
{120, 7.37142, 1.73123, 7.37142, 7.37144, 39.4784, 11.3926, 28.0392},
{121, 7.27163, 1.70387, 7.27163, 7.27164, 39.4784, 11.2108, 27.9481},
{122, 7.172, 1.67657, 7.172, 7.17201, 39.4784, 11.0323, 27.8567},
{123, 7.07214, 1.64934, 7.07214, 7.07215, 39.4784, 10.8498, 27.7633},
{124, 6.97256, 1.62217, 6.97256, 6.97256, 39.4784, 10.6723, 27.6698},
{125, 6.87225, 1.59506, 6.87225, 6.87226, 39.4784, 10.4924, 27.5777},
{126, 6.77228, 1.56802, 6.77228, 6.77229, 39.4784, 10.3157, 27.4841},
{127, 6.6726, 1.54105, 6.6726, 6.67261, 39.4784, 10.1362, 27.3861},
{128, 6.57216, 1.51414, 6.57216, 6.57216, 39.4784, 9.96542, 27.2958},
{129, 6.47157, 1.4873, 6.47157, 6.47158, 39.4784, 9.78488, 27.1997},
{130, 6.37236, 1.46053, 6.37236, 6.37237, 39.4784, 9.61062, 27.0976},
{131, 6.2719, 1.43383, 6.2719, 6.27191, 39.4784, 9.43301, 26.9995},
{132, 6.17237, 1.4072, 6.17237, 6.17238, 39.4784, 9.25832, 26.896},
{133, 6.07228, 1.38064, 6.07228, 6.07229, 39.4784, 9.08433, 26.7948},
{134, 5.97249, 1.35415, 5.97249, 5.9725, 39.4784, 8.90733, 26.6883},
{135, 5.8725, 1.32774, 5.8725, 5.87251, 39.4784, 8.73724, 26.5856},
{136, 5.77255, 1.30141, 5.77255, 5.77255, 39.4784, 8.56101, 26.4773},
{137, 5.6721, 1.27515, 5.6721, 5.67211, 39.4784, 8.38767, 26.3722},
{138, 5.5727, 1.24896, 5.5727, 5.5727, 39.4784, 8.21782, 26.2607},
{139, 5.47227, 1.22286, 5.47227, 5.47228, 39.4784, 8.04275, 26.1508},
{140, 5.37246, 1.19683, 5.37246, 5.37247, 39.4784, 7.872, 26.0377},
{141, 5.27263, 1.17089, 5.27263, 5.27263, 39.4784, 7.7029, 25.9238},
{142, 5.17241, 1.14502, 5.17241, 5.17242, 39.4784, 7.53262, 25.8098},
{143, 5.07251, 1.11924, 5.07251, 5.07252, 39.4784, 7.36308, 25.6919},
{144, 4.97275, 1.09355, 4.97275, 4.97275, 39.4784, 7.19435, 25.5715},
{145, 4.87241, 1.06794, 4.87241, 4.87242, 39.4784, 7.02531, 25.4528},
{146, 4.77256, 1.04242, 4.77256, 4.77257, 39.4784, 6.85703, 25.3286},
{147, 4.67275, 1.01698, 4.67275, 4.67275, 39.4784, 6.69052, 25.2031},
{148, 4.57223, 0.991619, 4.57219, 4.57217, 39.4784, 6.59197, 25.1324},

Values for p = 2.8, d = 5, adaptative mesh - 8 November 2011
i, kappa, Lambda(kappa), J(Lambda(kappa)), L2, L2gradient, Lp
Values for Lambda > 5

```
L811b = {{13, 19.4107, 5.76715, 19.4107, 39.4784, 195.298, 74.7454},
    {12, 19.9106, 6.05311, 19.9106, 39.4784, 224.5, 81.9799},
    {11, 20.4106, 6.35792, 20.4106, 39.4784, 255.334, 89.6217},
    {11, 20.9106, 6.6825, 20.9106, 39.4784, 288.201, 97.7729},
    {11, 21.4106, 7.02755, 21.4106, 39.4784, 323, 106.407},
    {13, 21.9106, 7.39374, 21.9106, 39.4784, 360.067, 115.609},
    {12, 22.4107, 7.7818, 22.4107, 39.4784, 398.851, 125.241},
    {10, 22.9107, 8.19242, 22.9107, 39.4784, 439.545, 135.35},
    {12, 23.4107, 8.62636, 23.4107, 39.4784, 483.203, 146.198},
    {12, 23.9107, 9.08434, 23.9107, 39.4784, 528.835, 157.539},
    {11, 24.4107, 9.56712, 24.4107, 39.4784, 576.721, 169.442},
    {11, 24.9107, 10.0755, 24.9107, 39.4784, 627.236, 182.001},
    {12, 25.4107, 10.6101, 25.4107, 39.4784, 680.466, 195.236},
    {11, 25.9107, 11.1719, 25.9107, 39.4784, 736.045, 209.057},
    {13, 26.4108, 11.7616, 26.4108, 39.4784, 794.755, 223.658},
    {12, 26.9108, 12.38, 26.9108, 39.4784, 855.923, 238.871},
    {12, 27.4108, 13.0279, 27.4108, 39.4784, 920.079, 254.828},
    {11, 27.9109, 13.7061, 27.9109, 39.4784, 987.053, 271.488},
    {10, 28.4109, 14.4155, 28.4109, 39.4784, 1056.98, 288.882},
    {11, 28.9109, 15.1569, 28.9109, 39.4784, 1130.56, 307.186},
    {11, 29.4109, 15.9311, 29.4109, 39.4784, 1207.11, 326.23},
    {16, 29.9109, 16.739, 29.9109, 39.4784, 1287.45, 346.217},
    {10, 30.411, 17.5814, 30.411, 39.4784, 1369.97, 366.748},
    {11, 30.911, 18.4593, 30.911, 39.4784, 1456.99, 388.399},
    {10, 31.4111, 19.3734, 31.4111, 39.4784, 1547.04, 410.804},
    {11, 31.9111, 20.3247, 31.9111, 39.4784, 1641.3, 434.256},
    {10, 32.4112, 21.3141, 32.4112, 39.4784, 1738.72, 458.494},
    {13, 32.9112, 22.3423, 32.9112, 39.4784, 1840.95, 483.93},
    {11, 33.4112, 23.4104, 33.4112, 39.4784, 1946.07, 510.086},
    {11, 33.9113, 24.5193, 33.9113, 39.4784, 2055.57, 537.332},
    {11, 34.4113, 25.6698, 34.4113, 39.4784, 2169.2, 565.605},
    {10, 34.9114, 26.863, 34.9114, 39.4784, 2286.67, 594.836},
    {12, 35.4114, 28.0996, 35.4114, 39.4784, 2409.39, 625.371},
    {10, 35.9115, 29.3808, 35.9115, 39.4784, 2535.23, 656.683},
    {11, 36.4115, 30.7072, 36.4115, 39.4784, 2666.56, 689.361},
    {10, 36.9116, 32.0801, 36.9116, 39.4784, 2801.7, 722.988},
    {10, 37.4116, 33.5004, 37.4116, 39.4784, 2941.87, 757.869},
    {10, 37.9117, 34.969, 37.9117, 39.4784, 3086.88, 793.952},
    {14, 38.4118, 36.4868, 38.4118, 39.4784, 3237.82, 831.509},
    {10, 38.9119, 38.0547, 38.9119, 39.4784, 3391.49, 869.748},
    {10, 39.4119, 39.6739, 39.4119, 39.4784, 3551.28, 909.51},
    {10, 39.912, 41.3452, 39.912, 39.4784, 3716.19, 950.547},
    {10, 40.4121, 43.07, 40.4121, 39.4784, 3886.5, 992.923},
    {11, 40.9122, 44.8487, 40.9122, 39.4784, 4062.55, 1036.73},
    {12, 41.4122, 46.6827, 41.4122, 39.4784, 4243.9, 1081.86},
    {10, 41.9123, 48.5729, 41.9123, 39.4784, 4429.57, 1128.06},
    {12, 42.4123, 50.5203, 42.4123, 39.4784, 4622.73, 1176.13},
    {11, 42.9124, 52.526, 42.9124, 39.4784, 4820.31, 1225.29},
    {10, 43.4125, 54.5911, 43.4125, 39.4784, 5023.65, 1275.89},
```

```
    {11, 43.9126, 56.7164, 43.9126, 39.4784, 5233.93, 1328.22},
    {10, 44.4126, 58.9032, 44.4126, 39.4784, 5449.14, 1381.77},
    {10, 44.9127, 61.1523, 44.9127, 39.4784, 5671.09, 1437},
    {10, 45.4127, 63.4648, 45.4127, 39.4784, 5899.32, 1493.79},
    {10, 45.9128, 65.8419, 45.9128, 39.4784, 6133.91, 1552.17},
    {10, 46.413, 68.2853, 46.413, 39.4784, 6375.23, 1612.22},
    {10, 46.913, 70.7948, 46.913, 39.4784, 6622.81, 1673.83},
    {10, 47.4131, 73.3721, 47.4131, 39.4784, 6877.16, 1737.12},
    {11, 47.9133, 76.0186, 47.9132, 39.4784, 7139.1, 1802.3},
    {10, 48.4134, 78.735, 48.4133, 39.4784, 7406.43, 1868.82},
    {10, 48.9134, 81.5223, 48.9134, 39.4784, 7681.44, 1937.26},
    {11, 49.4134, 84.3815, 49.4134, 39.4784, 7964.33, 2007.65},
    {10, 49.9135, 87.3142, 49.9135, 39.4784, 8253.04, 2079.5},
    {15, 50.4136, 90.3212, 50.4136, 39.4784, 8551.85, 2153.85},
    {10, 50.9136, 93.4036, 50.9136, 39.4784, 8853.99, 2229.04},
    {10, 51.4137, 96.5626, 51.4137, 39.4784, 9165.68, 2306.6},
    {11, 51.9137, 99.7996, 51.9137, 39.4784, 9485.98, 2386.31},
    {12, 52.4138, 103.115, 52.4138, 39.4784, 9813.8, 2467.88},
    {10, 52.9138, 106.511, 52.9138, 39.4784, 10147.3, 2550.88},
    {11, 53.414, 109.988, 53.414, 39.4784, 10491.7, 2636.57},
    {10, 53.9141, 113.548, 53.9141, 39.4784, 10842.1, 2723.77},
    {10, 54.4142, 117.191, 54.4142, 39.4784, 11201.6, 2813.22},
    {10, 54.9142, 120.919, 54.9142, 39.4784, 11569.4, 2904.75},
    {10, 55.4143, 124.732, 55.4143, 39.4784, 11945.7, 2998.39},
    {12, 55.9147, 128.634, 55.9147, 39.4784, 12333.1, 3094.78},
    {11, 56.4148, 132.623, 56.4148, 39.4784, 12726, 3192.55},
    {11, 56.9149, 136.702, 56.9149, 39.4784, 13128.3, 3292.66},
    {11, 57.4149, 140.87, 57.4149, 39.4784, 13539.6, 3395.01},
    {12, 57.9149, 145.131, 57.9149, 39.4784, 13960.7, 3499.81},
    {10, 58.4151, 149.485, 58.4151, 39.4784, 14388.6, 3606.28},
    {13, 58.9152, 153.932, 58.9152, 39.4784, 14829.9, 3716.09},
    {10, 59.4152, 158.475, 59.4152, 39.4784, 15275.7, 3827.03},
    {10, 59.9154, 163.115, 59.9154, 39.4784, 15733.8, 3941.03},
    {11, 60.4156, 167.853, 60.4155, 39.4784, 16202.8, 4057.74},
    {10, 60.9156, 172.69, 60.9156, 39.4784, 16678.8, 4176.19},
    {10, 61.4157, 177.626, 61.4157, 39.4784, 17165.9, 4297.4},
    {10, 61.9157, 182.665, 61.9157, 39.4784, 17663, 4421.09},
    {10, 62.4159, 187.807, 62.4159, 39.4784, 18170.7, 4547.44},
    {10, 62.916, 193.052, 62.916, 39.4784, 18688.2, 4676.21},
    {11, 63.4161, 198.402, 63.4161, 39.4784, 19217.6, 4807.96},
    {10, 63.9164, 203.861, 63.9164, 39.4784, 19755.4, 4941.76}};
```

Values for p = 2.80, d = 5

```
LL280err12 = {{0, 39.5342, 37.2825, 39.5235, 102.408, 39.4784, 2797.49, 702.923},
    {1, 39.2012, 38.9851, 39.2012, 106.416, 39.4784, 3484.39, 892.864},
    {2, 38.9513, 38.1803, 38.9512, 105.731, 39.4784, 3405.21, 873.163},
    {3, 38.7013, 37.3882, 38.7013, 105.212, 39.4784, 3327.1, 853.724},
    {4, 38.4514, 36.6087, 38.4514, 102.328, 39.4784, 3250.16, 834.577},
    {5, 38.2014, 35.8418, 38.2014, 100.588, 39.4784, 3174.44, 815.736},
    {6, 37.9514, 35.0872, 37.9514, 98.7819, 39.4784, 3099.96, 797.203},
    {7, 37.7014, 34.345, 37.7014, 97.0351, 39.4784, 3026.71, 778.976},
    {8, 37.4514, 33.615, 37.4514, 95.5633, 39.4784, 2954.66, 761.047},
    {9, 37.2015, 32.8971, 37.2015, 93.6679, 39.4784, 2883.82, 743.419},
```

```
{10, 36.9515, 32.1912, 36.9515, 92.3998, 39.4784, 2814.18, 726.087},
{11, 36.7015, 31.4971, 36.7015, 90.4877, 39.4784, 2745.64, 709.034},
{12, 36.4516, 30.8148, 36.4516, 88.1107, 39.4784, 2678.33, 692.284},
{13, 36.2016, 30.1441, 36.2016, 86.7039, 39.4784, 2612.16, 675.819},
{14, 35.9517, 29.4848, 35.9517, 85.5251, 39.4784, 2547.13, 659.634},
{15, 35.7017, 28.8372, 35.7017, 83.8008, 39.4784, 2483.01, 643.682},
{16, 35.4518, 28.2006, 35.4518, 82.1261, 39.4784, 2420.17, 628.047},
{17, 35.2018, 27.5753, 35.2018, 80.5221, 39.4784, 2358.35, 612.666},
{18, 34.9519, 26.961, 34.9519, 79.0778, 39.4784, 2297.62, 597.556},
{19, 34.702, 26.3577, 34.702, 77.3395, 39.4784, 2238.01, 582.724},
{20, 34.452, 25.765, 34.452, 75.9585, 39.4784, 2179.47, 568.158},
{21, 34.2021, 25.1828, 34.2021, 74.9191, 39.4784, 2121.98, 553.853},
{22, 33.9521, 24.6112, 33.9521, 72.9677, 39.4784, 2065.54, 539.808},
{23, 33.7022, 24.0501, 33.7022, 71.6733, 39.4784, 2010.13, 526.022},
{24, 33.4522, 23.4993, 33.4522, 70.3467, 39.4784, 1955.76, 512.492},
{25, 33.2023, 22.9586, 33.2022, 68.7996, 39.4784, 1902.36, 499.205},
{26, 32.9523, 22.4281, 32.9523, 67.8131, 39.4784, 1849.92, 486.159},
{27, 32.7024, 21.9076, 32.7024, 65.77, 39.4784, 1798.56, 473.379},
{28, 32.4524, 21.397, 32.4524, 64.5104, 39.4784, 1748.05, 460.811},
{29, 32.2025, 20.8961, 32.2025, 63.193, 39.4784, 1698.61, 448.509},
{30, 31.9525, 20.4048, 31.9525, 61.9034, 39.4784, 1649.99, 436.414},
{31, 31.7026, 19.923, 31.7026, 60.7184, 39.4784, 1602.4, 424.572},
{32, 31.4526, 19.4506, 31.4526, 59.4581, 39.4784, 1555.72, 412.959},
{33, 31.2026, 18.9876, 31.2026, 58.1929, 39.4784, 1509.9, 401.56},
{34, 30.9526, 18.5338, 30.9526, 56.8785, 39.4784, 1465.01, 390.393},
{35, 30.7026, 18.0889, 30.7026, 55.7493, 39.4784, 1421.03, 379.451},
{36, 30.4527, 17.6531, 30.4527, 54.6225, 39.4784, 1377.97, 368.736},
{37, 30.2027, 17.226, 30.2027, 53.4258, 39.4784, 1335.76, 358.236},
{38, 29.9527, 16.8078, 29.9527, 52.2629, 39.4784, 1294.42, 347.951},
{39, 29.7027, 16.3982, 29.7027, 51.091, 39.4784, 1253.94, 337.879},
{40, 29.4527, 15.9971, 29.4527, 49.9506, 39.4784, 1214.3, 328.016},
{41, 29.2028, 15.6045, 29.2028, 48.8822, 39.4784, 1175.46, 318.354},
{42, 28.9528, 15.2203, 28.9528, 47.8594, 39.4784, 1137.46, 308.901},
{43, 28.7028, 14.8443, 28.7028, 46.9304, 39.4784, 1100.21, 299.636},
{44, 28.4528, 14.4764, 28.4528, 45.9275, 39.4784, 1063.82, 290.582},
{45, 28.2028, 14.1164, 28.2028, 44.9137, 39.4784, 1028.21, 281.725},
{46, 27.9529, 13.7644, 27.9529, 43.9187, 39.4784, 993.387, 273.062},
{47, 27.7029, 13.4202, 27.7029, 42.9594, 39.4784, 959.308, 264.585},
{48, 27.4529, 13.0837, 27.4529, 41.9653, 39.4784, 925.991, 256.298},
{49, 27.2029, 12.7548, 27.2029, 41.0525, 39.4784, 893.417, 248.196},
{50, 26.9529, 12.4333, 26.9529, 40.1554, 39.4784, 861.593, 240.28},
{51, 26.703, 12.1193, 26.703, 39.236, 39.4784, 830.464, 232.538},
{52, 26.453, 11.8126, 26.453, 38.3538, 39.4784, 800.067, 224.978},
{53, 26.203, 11.513, 26.203, 37.5617, 39.4784, 770.377, 217.594},
{54, 25.953, 11.2206, 25.953, 36.7678, 39.4784, 741.38, 210.383},
{55, 25.703, 10.9351, 25.703, 35.9271, 39.4784, 713.057, 203.339},
{56, 25.453, 10.6565, 25.453, 35.1277, 39.4784, 685.418, 196.466},
{57, 25.203, 10.3847, 25.203, 34.3444, 39.4784, 658.44, 189.758},
{58, 24.9531, 10.1196, 24.953, 33.5827, 39.4784, 632.12, 183.214},
{59, 24.7031, 9.86111, 24.7031, 32.8263, 39.4784, 606.443, 176.83},
{60, 24.4531, 9.60914, 24.4531, 32.0991, 39.4784, 581.389, 170.602},
{61, 24.2031, 9.36355, 24.2031, 31.3873, 39.4784, 556.962, 164.529},
{62, 23.9531, 9.12427, 23.9531, 30.6874, 39.4784, 533.144, 158.609},
{63, 23.7031, 8.8912, 23.7031, 30.0143, 39.4784, 509.934, 152.84},
{64, 23.4531, 8.66425, 23.4531, 29.3555, 39.4784, 487.32, 147.22},
```

```
{65, 23.2031, 8.44332, 23.2031, 28.6868, 39.4784, 465.288, 141.745},
{66, 22.9531, 8.22831, 22.9531, 28.0773, 39.4784, 443.833, 136.415},
{67, 22.7031, 8.01916, 22.7031, 27.4576, 39.4784, 422.94, 131.224},
{68, 22.4531, 7.81575, 22.4531, 26.8527, 39.4784, 402.601, 126.172},
{69, 22.2031, 7.61801, 22.2031, 26.2681, 39.4784, 382.803, 121.255},
{70, 21.9531, 7.42583, 21.9531, 25.694, 39.4784, 363.544, 116.472},
{71, 21.7031, 7.23912, 21.7031, 25.1384, 39.4784, 344.81, 111.821},
{72, 21.4531, 7.05781, 21.4531, 24.5978, 39.4784, 326.592, 107.299},
{73, 21.2031, 6.8818, 21.2031, 24.0696, 39.4784, 308.881, 102.904},
{74, 20.9531, 6.711, 20.9531, 23.553, 39.4784, 291.666, 98.6322},
{75, 20.7031, 6.54534, 20.7031, 23.0538, 39.4784, 274.939, 94.4832},
{76, 20.4531, 6.38471, 20.4531, 22.5668, 39.4784, 258.69, 90.4538},
{77, 20.2031, 6.22905, 20.2031, 22.0942, 39.4784, 242.911, 86.542},
{78, 19.9531, 6.07826, 19.9531, 21.6349, 39.4784, 227.59, 82.7453},
{79, 19.7031, 5.93225, 19.7031, 21.1884, 39.4784, 212.72, 79.0618},
{80, 19.4531, 5.79096, 19.4531, 20.7551, 39.4784, 198.292, 75.4892},
{81, 19.2031, 5.65429, 19.2031, 20.3344, 39.4784, 184.296, 72.0253},
{82, 18.9531, 5.52216, 18.9531, 19.9264, 39.4784, 170.725, 68.6681},
{83, 18.7031, 5.3945, 18.7031, 19.5308, 39.4784, 157.569, 65.4156},
{84, 18.4531, 5.27123, 18.4531, 19.1476, 39.4784, 144.82, 62.2655},
{85, 18.2031, 5.15226, 18.2031, 18.7766, 39.4784, 132.469, 59.216},
{86, 17.9531, 5.03751, 17.9531, 18.4175, 39.4784, 120.509, 56.265},
{87, 17.7031, 4.92692, 17.7031, 18.0704, 39.4784, 108.931, 53.4107},
{88, 17.4531, 4.8204, 17.4531, 17.735, 39.4784, 97.7282, 50.6512},
{89, 17.2031, 4.71788, 17.2031, 17.4112, 39.4784, 86.8914, 47.9846},
{90, 16.9531, 4.61929, 16.9531, 17.0988, 39.4784, 76.4137, 45.4091},
{91, 16.7031, 4.52454, 16.7031, 16.7978, 39.4784, 66.2881, 42.9231},
{92, 16.4531, 4.43356, 16.4531, 16.5078, 39.4784, 56.5088, 40.5252},
{93, 16.2031, 4.34628, 16.2031, 16.2289, 39.4784, 47.081, 38.2167},
{94, 15.9531, 4.26263, 15.9531, 15.9608, 39.4784, 38.1202, 36.0259},
{95, 15.7031, 4.18248, 15.7031, 15.7032, 39.4784, 30.6099, 34.1926},
{96, 15.4531, 4.10482, 15.4531, 15.4529, 39.4784, 27.2168, 33.3646},
{97, 15.2031, 4.02751, 15.2031, 15.2031, 39.4784, 26.5, 33.1875},
{98, 14.9531, 3.95036, 14.9531, 14.9531, 39.4784, 25.9885, 33.0586},
{99, 14.7023, 3.87342, 14.7023, 14.7023, 39.4784, 25.4789, 32.9304},
{100, 14.453, 3.7967, 14.453, 14.453, 39.4784, 24.9812, 32.7985},
{101, 14.2017, 3.72019, 14.2017, 14.2017, 39.4784, 24.4739, 32.6686},
{102, 13.9531, 3.64393, 13.9531, 13.9531, 39.4784, 23.9743, 32.5299},
{103, 13.7018, 3.56787, 13.7018, 13.7019, 39.4784, 23.4768, 32.3978},
{104, 13.4524, 3.49206, 13.4524, 13.4524, 39.4784, 22.9666, 32.2542},
{105, 13.203, 3.41648, 13.203, 13.203, 39.4784, 22.48, 32.1144},
{106, 12.9529, 3.34114, 12.9529, 12.9529, 39.4784, 21.9799, 31.9706},
{107, 12.7025, 3.26603, 12.7025, 12.7025, 39.4784, 21.4947, 31.8299},
{108, 12.453, 3.19117, 12.453, 12.453, 39.4784, 20.9934, 31.6782},
{109, 12.2026, 3.11656, 12.2026, 12.2027, 39.4784, 20.5088, 31.532},
{110, 11.9526, 3.04221, 11.9526, 11.9526, 39.4784, 20.0106, 31.3777},
{111, 11.7029, 2.96811, 11.7029, 11.7029, 39.4784, 19.5347, 31.2259},
{112, 11.4533, 2.89428, 11.4533, 11.4533, 39.4784, 19.0411, 31.0651},
{113, 11.2026, 2.82071, 11.2026, 11.2026, 39.4784, 18.5647, 30.9108},
{114, 10.9526, 2.74742, 10.9526, 10.9526, 39.4784, 18.075, 30.7459},
{115, 10.7032, 2.67441, 10.7032, 10.7032, 39.4784, 17.5969, 30.5795},
{116, 10.4529, 2.60168, 10.4529, 10.4529, 39.4784, 17.1151, 30.4116},
{117, 10.2032, 2.52924, 10.2032, 10.2032, 39.4784, 16.6427, 30.2405},
{118, 9.95282, 2.45709, 9.95282, 9.95283, 39.4784, 16.1658, 30.0669},
{119, 9.70281, 2.38525, 9.70281, 9.70282, 39.4784, 15.6967, 29.8905},
```

```
{120, 9.45326, 2.31371, 9.45326, 9.45327, 39.4784, 15.2212, 29.7052},
{121, 9.20268, 2.24248, 9.20268, 9.20269, 39.4784, 14.7587, 29.5251},
{122, 8.95209, 2.17157, 8.95209, 8.95211, 39.4784, 14.2927, 29.339},
{123, 8.70313, 2.10101, 8.70313, 8.70314, 39.4784, 13.8193, 29.1375},
{124, 8.45333, 2.03077, 8.45333, 8.45333, 39.4784, 13.3621, 28.9413}};

LL280err12moinspremierterme =
 {{1, 39.2012, 38.9851, 39.2012, 106.416, 39.4784, 3484.39, 892.864},
  {2, 38.9513, 38.1803, 38.9512, 105.731, 39.4784, 3405.21, 873.163},
  {3, 38.7013, 37.3882, 38.7013, 105.212, 39.4784, 3327.1, 853.724},
  {4, 38.4514, 36.6087, 38.4514, 102.328, 39.4784, 3250.16, 834.577},
  {5, 38.2014, 35.8418, 38.2014, 100.588, 39.4784, 3174.44, 815.736},
  {6, 37.9514, 35.0872, 37.9514, 98.7819, 39.4784, 3099.96, 797.203},
  {7, 37.7014, 34.345, 37.7014, 97.0351, 39.4784, 3026.71, 778.976},
  {8, 37.4514, 33.615, 37.4514, 95.5633, 39.4784, 2954.66, 761.047},
  {9, 37.2015, 32.8971, 37.2015, 93.6679, 39.4784, 2883.82, 743.419},
  {10, 36.9515, 32.1912, 36.9515, 92.3998, 39.4784, 2814.18, 726.087},
  {11, 36.7015, 31.4971, 36.7015, 90.4877, 39.4784, 2745.64, 709.034},
  {12, 36.4516, 30.8148, 36.4516, 88.1107, 39.4784, 2678.33, 692.284},
  {13, 36.2016, 30.1441, 36.2016, 86.7039, 39.4784, 2612.16, 675.819},
  {14, 35.9517, 29.4848, 35.9517, 85.5251, 39.4784, 2547.13, 659.634},
  {15, 35.7017, 28.8372, 35.7017, 83.8008, 39.4784, 2483.01, 643.682},
  {16, 35.4518, 28.2006, 35.4518, 82.1261, 39.4784, 2420.17, 628.047},
  {17, 35.2018, 27.5753, 35.2018, 80.5221, 39.4784, 2358.35, 612.666},
  {18, 34.9519, 26.961, 34.9519, 79.0778, 39.4784, 2297.62, 597.556},
  {19, 34.702, 26.3577, 34.702, 77.3395, 39.4784, 2238.01, 582.724},
  {20, 34.452, 25.765, 34.452, 75.9585, 39.4784, 2179.47, 568.158},
  {21, 34.2021, 25.1828, 34.2021, 74.9191, 39.4784, 2121.98, 553.853},
  {22, 33.9521, 24.6112, 33.9521, 72.9677, 39.4784, 2065.54, 539.808},
  {23, 33.7022, 24.0501, 33.7022, 71.6733, 39.4784, 2010.13, 526.022},
  {24, 33.4522, 23.4993, 33.4522, 70.3467, 39.4784, 1955.76, 512.492},
  {25, 33.2023, 22.9586, 33.2022, 68.7996, 39.4784, 1902.36, 499.205},
  {26, 32.9523, 22.4281, 32.9523, 67.8131, 39.4784, 1849.92, 486.159},
  {27, 32.7024, 21.9076, 32.7024, 65.77, 39.4784, 1798.56, 473.379},
  {28, 32.4524, 21.397, 32.4524, 64.5104, 39.4784, 1748.05, 460.811},
  {29, 32.2025, 20.8961, 32.2025, 63.193, 39.4784, 1698.61, 448.509},
  {30, 31.9525, 20.4048, 31.9525, 61.9034, 39.4784, 1649.99, 436.414},
  {31, 31.7026, 19.923, 31.7026, 60.7184, 39.4784, 1602.4, 424.572},
  {32, 31.4526, 19.4506, 31.4526, 59.4581, 39.4784, 1555.72, 412.959},
  {33, 31.2026, 18.9876, 31.2026, 58.1929, 39.4784, 1509.9, 401.56},
  {34, 30.9526, 18.5338, 30.9526, 56.8785, 39.4784, 1465.01, 390.393},
  {35, 30.7026, 18.0889, 30.7026, 55.7493, 39.4784, 1421.03, 379.451},
  {36, 30.4527, 17.6531, 30.4527, 54.6225, 39.4784, 1377.97, 368.736},
  {37, 30.2027, 17.226, 30.2027, 53.4258, 39.4784, 1335.76, 358.236},
  {38, 29.9527, 16.8078, 29.9527, 52.2629, 39.4784, 1294.42, 347.951},
  {39, 29.7027, 16.3982, 29.7027, 51.091, 39.4784, 1253.94, 337.879},
  {40, 29.4527, 15.9971, 29.4527, 49.9506, 39.4784, 1214.3, 328.016},
  {41, 29.2028, 15.6045, 29.2028, 48.8822, 39.4784, 1175.46, 318.354},
  {42, 28.9528, 15.2203, 28.9528, 47.8594, 39.4784, 1137.46, 308.901},
  {43, 28.7028, 14.8443, 28.7028, 46.9304, 39.4784, 1100.21, 299.636},
  {44, 28.4528, 14.4764, 28.4528, 45.9275, 39.4784, 1063.82, 290.582},
  {45, 28.2028, 14.1164, 28.2028, 44.9137, 39.4784, 1028.21, 281.725},
  {46, 27.9529, 13.7644, 27.9529, 43.9187, 39.4784, 993.387, 273.062},
  {47, 27.7029, 13.4202, 27.7029, 42.9594, 39.4784, 959.308, 264.585},
  {48, 27.4529, 13.0837, 27.4529, 41.9653, 39.4784, 925.991, 256.298},
```

```
{49, 27.2029, 12.7548, 27.2029, 41.0525, 39.4784, 893.417, 248.196},
{50, 26.9529, 12.4333, 26.9529, 40.1554, 39.4784, 861.593, 240.28},
{51, 26.703, 12.1193, 26.703, 39.236, 39.4784, 830.464, 232.538},
{52, 26.453, 11.8126, 26.453, 38.3538, 39.4784, 800.067, 224.978},
{53, 26.203, 11.513, 26.203, 37.5617, 39.4784, 770.377, 217.594},
{54, 25.953, 11.2206, 25.953, 36.7678, 39.4784, 741.38, 210.383},
{55, 25.703, 10.9351, 25.703, 35.9271, 39.4784, 713.057, 203.339},
{56, 25.453, 10.6565, 25.453, 35.1277, 39.4784, 685.418, 196.466},
{57, 25.203, 10.3847, 25.203, 34.3444, 39.4784, 658.44, 189.758},
{58, 24.9531, 10.1196, 24.953, 33.5827, 39.4784, 632.12, 183.214},
{59, 24.7031, 9.86111, 24.7031, 32.8263, 39.4784, 606.443, 176.83},
{60, 24.4531, 9.60914, 24.4531, 32.0991, 39.4784, 581.389, 170.602},
{61, 24.2031, 9.36355, 24.2031, 31.3873, 39.4784, 556.962, 164.529},
{62, 23.9531, 9.12427, 23.9531, 30.6874, 39.4784, 533.144, 158.609},
{63, 23.7031, 8.8912, 23.7031, 30.0143, 39.4784, 509.934, 152.84},
{64, 23.4531, 8.66425, 23.4531, 29.3555, 39.4784, 487.32, 147.22},
{65, 23.2031, 8.44332, 23.2031, 28.6868, 39.4784, 465.288, 141.745},
{66, 22.9531, 8.22831, 22.9531, 28.0773, 39.4784, 443.833, 136.415},
{67, 22.7031, 8.01916, 22.7031, 27.4576, 39.4784, 422.94, 131.224},
{68, 22.4531, 7.81575, 22.4531, 26.8527, 39.4784, 402.601, 126.172},
{69, 22.2031, 7.61801, 22.2031, 26.2681, 39.4784, 382.803, 121.255},
{70, 21.9531, 7.42583, 21.9531, 25.694, 39.4784, 363.544, 116.472},
{71, 21.7031, 7.23912, 21.7031, 25.1384, 39.4784, 344.81, 111.821},
{72, 21.4531, 7.05781, 21.4531, 24.5978, 39.4784, 326.592, 107.299},
{73, 21.2031, 6.8818, 21.2031, 24.0696, 39.4784, 308.881, 102.904},
{74, 20.9531, 6.711, 20.9531, 23.553, 39.4784, 291.666, 98.6322},
{75, 20.7031, 6.54534, 20.7031, 23.0538, 39.4784, 274.939, 94.4832},
{76, 20.4531, 6.38471, 20.4531, 22.5668, 39.4784, 258.69, 90.4538},
{77, 20.2031, 6.22905, 20.2031, 22.0942, 39.4784, 242.911, 86.542},
{78, 19.9531, 6.07826, 19.9531, 21.6349, 39.4784, 227.59, 82.7453},
{79, 19.7031, 5.93225, 19.7031, 21.1884, 39.4784, 212.72, 79.0618},
{80, 19.4531, 5.79096, 19.4531, 20.7551, 39.4784, 198.292, 75.4892},
{81, 19.2031, 5.65429, 19.2031, 20.3344, 39.4784, 184.296, 72.0253},
{82, 18.9531, 5.52216, 18.9531, 19.9264, 39.4784, 170.725, 68.6681},
{83, 18.7031, 5.3945, 18.7031, 19.5308, 39.4784, 157.569, 65.4156},
{84, 18.4531, 5.27123, 18.4531, 19.1476, 39.4784, 144.82, 62.2655},
{85, 18.2031, 5.15226, 18.2031, 18.7766, 39.4784, 132.469, 59.216},
{86, 17.9531, 5.03751, 17.9531, 18.4175, 39.4784, 120.509, 56.265},
{87, 17.7031, 4.92692, 17.7031, 18.0704, 39.4784, 108.931, 53.4107},
{88, 17.4531, 4.8204, 17.4531, 17.735, 39.4784, 97.7282, 50.6512},
{89, 17.2031, 4.71788, 17.2031, 17.4112, 39.4784, 86.8914, 47.9846},
{90, 16.9531, 4.61929, 16.9531, 17.0988, 39.4784, 76.4137, 45.4091},
{91, 16.7031, 4.52454, 16.7031, 16.7978, 39.4784, 66.2881, 42.9231},
{92, 16.4531, 4.43356, 16.4531, 16.5078, 39.4784, 56.5088, 40.5252},
{93, 16.2031, 4.34628, 16.2031, 16.2289, 39.4784, 47.081, 38.2167},
{94, 15.9531, 4.26263, 15.9531, 15.9608, 39.4784, 38.1202, 36.0259},
{95, 15.7031, 4.18248, 15.7031, 15.7032, 39.4784, 30.6099, 34.1926},
{96, 15.4531, 4.10482, 15.4531, 15.4529, 39.4784, 27.2168, 33.3646},
{97, 15.2031, 4.02751, 15.2031, 15.2031, 39.4784, 26.5, 33.1875},
{98, 14.9531, 3.95036, 14.9531, 14.9531, 39.4784, 25.9885, 33.0586},
{99, 14.7023, 3.87342, 14.7023, 14.7023, 39.4784, 25.4789, 32.9304},
{100, 14.453, 3.7967, 14.453, 14.453, 39.4784, 24.9812, 32.7985},
{101, 14.2017, 3.72019, 14.2017, 14.2017, 39.4784, 24.4739, 32.6686},
{102, 13.9531, 3.64393, 13.9531, 13.9531, 39.4784, 23.9743, 32.5299},
{103, 13.7018, 3.56787, 13.7018, 13.7019, 39.4784, 23.4768, 32.3978},
```

```
{104, 13.4524, 3.49206, 13.4524, 13.4524, 39.4784, 22.9666, 32.2542},
{105, 13.203, 3.41648, 13.203, 13.203, 39.4784, 22.48, 32.1144},
{106, 12.9529, 3.34114, 12.9529, 12.9529, 39.4784, 21.9799, 31.9706},
{107, 12.7025, 3.26603, 12.7025, 12.7025, 39.4784, 21.4947, 31.8299},
{108, 12.453, 3.19117, 12.453, 12.453, 39.4784, 20.9934, 31.6782},
{109, 12.2026, 3.11656, 12.2026, 12.2027, 39.4784, 20.5088, 31.532},
{110, 11.9526, 3.04221, 11.9526, 11.9526, 39.4784, 20.0106, 31.3777},
{111, 11.7029, 2.96811, 11.7029, 11.7029, 39.4784, 19.5347, 31.2259},
{112, 11.4533, 2.89428, 11.4533, 11.4533, 39.4784, 19.0411, 31.0651},
{113, 11.2026, 2.82071, 11.2026, 11.2026, 39.4784, 18.5647, 30.9108},
{114, 10.9526, 2.74742, 10.9526, 10.9526, 39.4784, 18.075, 30.7459},
{115, 10.7032, 2.67441, 10.7032, 10.7032, 39.4784, 17.5969, 30.5795},
{116, 10.4529, 2.60168, 10.4529, 10.4529, 39.4784, 17.1151, 30.4116},
{117, 10.2032, 2.52924, 10.2032, 10.2032, 39.4784, 16.6427, 30.2405},
{118, 9.95282, 2.45709, 9.95282, 9.95283, 39.4784, 16.1658, 30.0669},
{119, 9.70281, 2.38525, 9.70281, 9.70282, 39.4784, 15.6967, 29.8905},
{120, 9.45326, 2.31371, 9.45326, 9.45327, 39.4784, 15.2212, 29.7052},
{121, 9.20268, 2.24248, 9.20268, 9.20269, 39.4784, 14.7587, 29.5251},
{122, 8.95209, 2.17157, 8.95209, 8.95211, 39.4784, 14.2927, 29.339},
{123, 8.70313, 2.10101, 8.70313, 8.70314, 39.4784, 13.8193, 29.1375},
{124, 8.45333, 2.03077, 8.45333, 8.45333, 39.4784, 13.3621, 28.9413}};
```

Values for p = 3.15, d = 5

```
LL315errmoins10 =
  {{0, 15.601`, 2.88261`, 15.6011`, 15.7562`, 39.4784`, 93.307`, 58.7208`},
   {44, 14.7518`, 2.65951`, 14.7518`, 14.7518`, 39.4784`, 23.4414`, 30.2163`},
   {43, 14.7718`, 2.66392`, 14.7718`, 14.7718`, 39.4784`, 23.4796`, 30.2306`},
   {42, 14.7915`, 2.66832`, 14.7915`, 14.7915`, 39.4784`, 23.5266`, 30.2488`},
   {41, 14.8118`, 2.67273`, 14.8118`, 14.8118`, 39.4784`, 23.5571`, 30.2589`},
   {40, 14.8294`, 2.67713`, 14.8294`, 14.8295`, 39.4784`, 23.6207`, 30.2898`},
   {39, 14.8518`, 2.68152`, 14.8518`, 14.8518`, 39.4784`, 23.6367`, 30.2879`},
   {38, 14.8717`, 2.68594`, 14.8717`, 14.8717`, 39.4784`, 23.6752`, 30.3022`},
   {37, 14.8918`, 2.69035`, 14.8918`, 14.8918`, 39.4784`, 23.7362`, 30.3243`},
   {36, 14.9118`, 2.69477`, 14.9118`, 14.9118`, 39.4784`, 23.9125`, 30.3889`},
   {35, 14.9318`, 2.69918`, 14.9318`, 14.9317`, 39.4784`, 24.4243`, 30.5766`},
   {34, 14.9518`, 2.70362`, 14.9518`, 14.9518`, 39.4784`, 25.3841`, 30.9297`},
   {33, 14.9718`, 2.7081`, 14.9718`, 14.9721`, 39.4784`, 26.6994`, 31.4155`},
   {32, 14.9918`, 2.71264`, 14.9918`, 14.9926`, 39.4784`, 28.2314`, 31.9842`},
   {31, 15.0118`, 2.71723`, 15.0118`, 15.0134`, 39.4784`, 29.8905`, 32.6035`},
   {30, 15.0318`, 2.72189`, 15.0318`, 15.0344`, 39.4784`, 31.624`, 33.2543`},
   {29, 15.0518`, 2.72661`, 15.0518`, 15.0557`, 39.4784`, 33.4037`, 33.9265`},
   {28, 15.0718`, 2.73138`, 15.0718`, 15.0772`, 39.4784`, 35.2241`, 34.6183`},
   {27, 15.0918`, 2.73622`, 15.0918`, 15.0991`, 39.4784`, 37.078`, 35.327`},
   {26, 15.1118`, 2.74112`, 15.1118`, 15.1212`, 39.4784`, 38.9579`, 36.0501`},
   {25, 15.1318`, 2.74609`, 15.1318`, 15.1436`, 39.4784`, 40.8652`, 36.7881`},
   {24, 15.1518`, 2.75112`, 15.1518`, 15.1662`, 39.4784`, 42.7979`, 37.5406`},
   {23, 15.1718`, 2.75621`, 15.1718`, 15.1892`, 39.4784`, 44.7567`, 38.3079`},
   {22, 15.1918`, 2.76137`, 15.1918`, 15.2124`, 39.4784`, 46.7429`, 39.0906`},
   {21, 15.2118`, 2.7666`, 15.2118`, 15.236`, 39.4784`, 48.7555`, 39.8886`},
   {20, 15.2318`, 2.77189`, 15.2318`, 15.2598`, 39.4784`, 50.7955`, 40.7023`},
   {19, 15.2518`, 2.77726`, 15.2518`, 15.2839`, 39.4784`, 52.8663`, 41.5333`},
   {18, 15.2718`, 2.78269`, 15.2718`, 15.3084`, 39.4784`, 54.9642`, 42.3804`},
   {17, 15.2918`, 2.78819`, 15.2918`, 15.3331`, 39.4784`, 57.0883`, 43.2432`},
```

```
{16, 15.3118`, 2.79376`, 15.3118`, 15.3581`, 39.4784`, 59.2442`, 44.1243`},
{15, 15.3318`, 2.79941`, 15.3318`, 15.3835`, 39.4784`, 61.4305`, 45.0232`},
{14, 15.3518`, 2.80512`, 15.3518`, 15.4092`, 39.4784`, 63.646`, 45.9396`},
{13, 15.3718`, 2.81092`, 15.3718`, 15.4352`, 39.4784`, 65.8921`, 46.8744`},
{12, 15.3918`, 2.81678`, 15.3918`, 15.4615`, 39.4784`, 68.1672`, 47.827`},
{11, 15.4118`, 2.82272`, 15.4118`, 15.4882`, 39.4784`, 70.4766`, 48.7997`},
{10, 15.4318`, 2.82874`, 15.4318`, 15.5151`, 39.4784`, 72.8158`, 49.7911`},
{9, 15.4518`, 2.83483`, 15.4518`, 15.5425`, 39.4784`, 75.1838`, 50.8006`},
{8, 15.4718`, 2.84101`, 15.4718`, 15.5701`, 39.4784`, 77.5903`, 51.8329`},
{7, 15.4918`, 2.84726`, 15.4918`, 15.5981`, 39.4784`, 80.0276`, 52.8847`},
{6, 15.5118`, 2.85359`, 15.5118`, 15.6265`, 39.4784`, 82.4945`, 53.9557`},
{5, 15.5318`, 2.86001`, 15.5318`, 15.6552`, 39.4784`, 85.0007`, 55.0504`},
{4, 15.5518`, 2.8665`, 15.5518`, 15.6842`, 39.4784`, 87.539`, 56.1658`},
{3, 15.5718`, 2.87308`, 15.5718`, 15.7137`, 39.4784`, 90.1099`, 57.3023`},
{2, 15.5918`, 2.87974`, 15.5918`, 15.7434`, 39.4784`, 92.7179`, 58.4623`},
{1, 15.6118`, 2.88649`, 15.6118`, 15.7736`, 39.4784`, 95.3604`, 59.6447`},
{0, 15.632`, 2.89312`, 15.632`, 15.8032`, 39.4784`, 97.3725`, 60.5446`},
{1, 15.7008, 2.91761, 15.7008, 15.9125, 39.4784, 107.559, 65.1939},
{2, 15.8008, 2.95471, 15.8008, 16.0777, 39.4784, 122.174, 72.0349},
{3, 15.9008, 2.99423, 15.9008, 16.2533, 39.4784, 137.801, 79.5725},
{4, 16.0008, 3.03633, 16.0008, 16.4399, 39.4784, 154.508, 87.8768},
{5, 16.1008, 3.08116, 16.1008, 16.638, 39.4784, 172.364, 97.0216},
{6, 16.2008, 3.1289, 16.2009, 16.8485, 39.4784, 191.438, 107.086},
{7, 16.3009, 3.17972, 16.3009, 17.0718, 39.4784, 211.809, 118.161},
{8, 16.4009, 3.23381, 16.4009, 17.3089, 39.4784, 233.556, 130.341},
{9, 16.5009, 3.29137, 16.5009, 17.5604, 39.4784, 256.762, 143.728},
{10, 16.6009, 3.35261, 16.6009, 17.8271, 39.4784, 281.515, 158.437},
{11, 16.7009, 3.41775, 16.7009, 18.1098, 39.4784, 307.91, 174.591},
{12, 16.8009, 3.48703, 16.8009, 18.4094, 39.4784, 336.045, 192.322},
{13, 16.9009, 3.56068, 16.9009, 18.7267, 39.4784, 366.017, 211.774},
{14, 17.0009, 3.63897, 17.0009, 19.0629, 39.4784, 397.937, 233.103},
{15, 17.1009, 3.72217, 17.1009, 19.4186, 39.4784, 431.921, 256.481},
{16, 17.2009, 3.81057, 17.2009, 19.7959, 39.4784, 468.078, 282.086},
{17, 17.3009, 3.90446, 17.3009, 20.1954, 39.4784, 506.539, 310.118},
{18, 17.4009, 4.00415, 17.4009, 20.6158, 39.4784, 547.422, 340.785},
{19, 17.501, 4.10999, 17.501, 21.0673, 39.4784, 590.871, 374.322},
{20, 17.601, 4.22231, 17.601, 21.5396, 39.4784, 637.028, 410.977},
{21, 17.701, 4.34147, 17.701, 22.031, 39.4784, 686.035, 451.015},
{22, 17.801, 4.46786, 17.801, 22.5537, 39.4784, 738.057, 494.732},
{23, 17.9011, 4.60189, 17.9011, 23.1072, 39.4784, 793.278, 542.457},
{24, 18.0011, 4.74397, 18.0011, 23.6947, 39.4784, 851.843, 594.508},
{25, 18.1011, 4.89453, 18.1011, 24.3195, 39.4784, 913.894, 651.212},
{26, 18.2012, 5.05405, 18.2012, 25.0091, 39.4784, 979.701, 713.034},
{27, 18.3012, 5.22299, 18.3012, 25.6815, 39.4784, 1049.38, 780.322},
{28, 18.4012, 5.40186, 18.4012, 26.4028, 39.4784, 1123.18, 853.567},
{29, 18.5012, 5.59117, 18.5012, 27.2018, 39.4784, 1201.31, 933.239},
{30, 18.6013, 5.79151, 18.6013, 28.0053, 39.4784, 1284.01, 1019.9},
{31, 18.7013, 6.00344, 18.7013, 28.7759, 39.4784, 1371.48, 1114.07},
{32, 18.8014, 6.22762, 18.8014, 29.8114, 39.4784, 1464.15, 1216.53},
{33, 18.9015, 6.46467, 18.9015, 30.6925, 39.4784, 1562.02, 1327.67},
{34, 19.0015, 6.71516, 19.0016, 31.6744, 39.4784, 1665.41, 1448.25},
{35, 19.1016, 6.97979, 19.1016, 32.683, 39.4784, 1774.69, 1579.1},
{36, 19.2016, 7.25932, 19.2016, 33.8025, 39.4784, 1890.11, 1720.97},
{37, 19.3017, 7.55451, 19.3017, 34.8781, 39.4784, 2012.03, 1874.81},
{38, 19.4018, 7.86619, 19.4018, 36.2161, 39.4784, 2140.75, 2041.49},
```

```
{39, 19.5018, 8.19503, 19.5018, 37.4406, 39.4784, 2276.47, 2221.85},
{40, 19.6019, 8.54214, 19.6019, 38.6699, 39.4784, 2419.94, 2417.46},
{41, 19.702, 8.90823, 19.702, 40.0603, 39.4784, 2571.11, 2628.93},
{42, 19.802, 9.29424, 19.802, 41.4542, 39.4784, 2730.45, 2857.58},
{43, 19.9021, 9.70142, 19.9021, 43.1469, 39.4784, 2898.7, 3105.19},
{44, 20.0022, 10.1305, 20.0022, 44.7283, 39.4784, 3075.89, 3372.62},
{45, 20.1023, 10.5826, 20.1023, 46.318, 39.4784, 3262.77, 3661.86},
{46, 20.2025, 11.059, 20.2025, 48.016, 39.4784, 3459.7, 3974.33},
{47, 20.3025, 11.5607, 20.3025, 49.6946, 39.4784, 3666.51, 4310.78},
{48, 20.4025, 12.0887, 20.4025, 51.5377, 39.4784, 3884.45, 4674.17},
{49, 20.5026, 12.6445, 20.5026, 53.5031, 39.4784, 4114, 5066.42},
{50, 20.6028, 13.2298, 20.6028, 55.6776, 39.4784, 4356.02, 5490.23},
{51, 20.7028, 13.8453, 20.7028, 57.9492, 39.4784, 4609.97, 5945.94},
{52, 20.8029, 14.4927, 20.8029, 60.1981, 39.4784, 4877.2, 6437.21},
{53, 20.903, 15.1736, 20.903, 62.0517, 39.4784, 5158.73, 6967.4},
{54, 21.003, 15.8893, 21.003, 64.8792, 39.4784, 5453.6, 7536.23},
{55, 21.1032, 16.6422, 21.1032, 67.3545, 39.4784, 5765.05, 8151.49},
{56, 21.2035, 17.4333, 21.2035, 70.451, 39.4784, 6092.61, 8814.17},
{57, 21.3035, 18.2642, 21.3035, 73.449, 39.4784, 6435.01, 9523.58},
{58, 21.4036, 19.1369, 21.4036, 76.0624, 39.4784, 6795.28, 10287.7},
{59, 21.5039, 20.0538, 21.5039, 79.0502, 39.4784, 7175.2, 11112.6},
{60, 21.6039, 21.0158, 21.6039, 82.1855, 39.4784, 7571.55, 11993.6},
{61, 21.7041, 22.0267, 21.7041, 85.4541, 39.4784, 7989.39, 12944.2},
{62, 21.8042, 23.087, 21.8042, 88.8838, 39.4784, 8426.38, 13961.6},
{63, 21.9044, 24.1993, 21.9044, 92.1989, 39.4784, 8885.83, 15056.1},
{64, 22.0046, 25.3663, 22.0046, 96.6248, 39.4784, 9368.1, 16231.6},
{65, 22.1048, 26.5913, 22.1048, 101.162, 39.4784, 9874.77, 17495.2},
{66, 22.2051, 27.8752, 22.2051, 103.645, 39.4784, 10405.3, 18848.6},
{67, 22.3051, 29.2196, 22.3051, 107.319, 39.4784, 10959.4, 20294.7},
{68, 22.4051, 30.6288, 22.4051, 111.421, 39.4784, 11540.2, 21844.9},
{69, 22.5053, 32.1055, 22.5053, 116.357, 39.4784, 12150.5, 23510.4},
{70, 22.6056, 33.6548, 22.6056, 120.838, 39.4784, 12791.8, 25300.1},
{71, 22.7057, 35.2767, 22.7057, 128.15, 39.4784, 13459.7, 27205.9},
{72, 22.8061, 36.9764, 22.8061, 134.657, 39.4784, 14162.9, 29257.5},
{73, 22.9062, 38.754, 22.9062, 136.936, 39.4784, 14895.7, 31443.1},
{74, 23.0063, 40.614, 23.0063, 142.39, 39.4784, 15664.5, 33786.3},
{75, 23.1064, 42.5598, 23.1064, 148.312, 39.4784, 16466.8, 36286.3},
{76, 23.2064, 44.5962, 23.2064, 153.194, 39.4784, 17307.3, 38962},
{77, 23.3067, 46.7285, 23.3067, 159.451, 39.4784, 18189.8, 41833.2},
{78, 23.4068, 48.9575, 23.4068, 164.522, 39.4784, 19108.8, 44888.3},
{79, 23.5071, 51.2917, 23.5071, 171.339, 39.4784, 20073.9, 48166.3}};
```

Qualitatives properties of the branches

Definitions

```
ac[d_] := (d - 2)/2

Theta[p_, d_] := d (p - 2)/(2 p)

S[d_] := 2 π^(d/2) / Gamma[d/2]

μFS[d_, p_] := 4 (d - 1)/(p^2 - 4)

Cstar[p_, d_, λ_, θ_] :=
 S[d]^(-(p-2)/p) ((p - 2)^2/(2 + (2 θ - 1) p))^((p-2)/(2 p)) ((2 + (2 θ - 1) p)/(2 p θ))^θ (4/(p + 2))^((6-p)/(2 p)) (Gamma[2/(p-2) + 1/2]/(Sqrt[π] Gamma[2/(p-2)]))^((p-2)/p) λ^((p-2)/(2 p) - θ)

Lambda[θ_, Lst_, n_] := θ Abs[Lst[[n]][[3]]] - (1 - θ) Lst[[n]][[6]]/Lst[[n]][[5]]

Energy[p_, θ_, Lst_, n_] :=
 ((Lst[[n]][[6]] + Lambda[θ, Lst, n] Lst[[n]][[5]])^θ Lst[[n]][[5]]^(1-θ)) /
  Lst[[n]][[7]]^(2/p)

PlotTable[Lst_, n1_, n2_] := ListLinePlot[
  Table[{Abs[Lst[[n]][[n1]]], Lst[[n]][[n2]]}, {n, 1, Length[Lst]}],
  PlotStyle → Directive[Black, Thickness[0.005]]]

Visualize = $DisplayFunction;

Constante = 2^(-(p-2)/p) /. p → 2.8;
θ0 = Theta[2.8, 5];
θList = {0.72, 0.75, 0.8, 0.85, 0.9, 0.95, 1};
PlotTableEnergyTheta[p_, θ_, Lst_] :=
 ListLinePlot[Table[{Lambda[θ, Lst, n], Energy[p, θ, Lst, n]}, {n, 1, Length[Lst]}],
  PlotStyle → Directive[Black, Thickness[0.005]]]

Lambdabis[θ_, Lst_, n_] := θ Abs[Lst[[n]][[3]]] - (1 - θ) Lst[[n]][[7]]/Lst[[n]][[6]]

Energybis[p_, θ_, Lst_, n_] :=
 ((Lst[[n]][[7]] + Lambdabis[θ, Lst, n] Lst[[n]][[6]])^θ Lst[[n]][[6]]^(1-θ)) /
  Lst[[n]][[8]]^(2/p)

Lambdastar1[p_, θ_,  mu_] := θ mu - (1 - θ) mu (p - 2)/(p + 2);

LambdaFS[p_, θ_, d_] := Lambdastar1[p, θ, μFS[d, p]]

PlotTableEnergyThetabis[p_, θ_, Lst_] := ListLinePlot[
  Table[{Lambdabis[θ, Lst, n], Energybis[p, θ, Lst, n]}, {n, 1, Length[Lst]}],
  PlotStyle → Directive[Thickness[0.005], Black]]
```

```
LL[h_, l_] := ListLinePlot[{{0, h}, {l, h}}, PlotStyle → {Thickness[0.001], Black}]
HH[h_, l_] := ListLinePlot[{{l, 0}, {l, h}}, PlotStyle → {Thickness[0.001], Black}]
```

Catrina and Wang asymptotics and Gagliardo-Nirenberg inequalities

```
K[θ_, p_] := ((p - 2)^2 / (2 + (2 θ - 1) p))^((p-2)/(2 p)) ((2 + (2 θ - 1) p) / (2 p θ))^θ (4 / (p + 2))^((6-p)/(2 p))
    (Gamma[2/(p-2) + 1/2] / (Sqrt[π] Gamma[2/(p-2)]))^((p-2)/p)
```

```
F[a_, p_, d_, rmax_, ϵ_, DF_, PR_] := Module[
  {M = Evaluate[u[s] /. NDSolve[{v'[r] + (d - 1) v[r]/r + a/Theta[p, d] Abs[u[r]]^(p-2) u[r] -
          (1 - Theta[p, d])/Theta[p, d] u[r] == 0, u'[r] == v[r], v[ϵ] == -(a + Theta[p, d] - 1)/Theta[p, d] ϵ/d,
        u[ϵ] == 1 - (a + Theta[p, d] - 1)/Theta[p, d] ϵ^2/(2 d)}, {u, v}, {r, ϵ, rmax}]]},
  Plot[M, {s, ϵ, rmax}, DisplayFunction → DF, PlotRange → PR]]
```

```
H[a_, p_, d_, rmax_, ϵ_] := Log[1 + u[rmax]^2 + v[rmax]^2] /.
  NDSolve[{v'[r] + (d - 1) v[r]/r + a/Theta[p, d] Abs[u[r]]^(p-2) u[r] - (1 - Theta[p, d])/Theta[p, d] u[r] ==
       0, u'[r] == v[r], v[ϵ] == -(a + Theta[p, d] - 1)/Theta[p, d] ϵ/d,
      u[ϵ] == 1 - (a + Theta[p, d] - 1)/Theta[p, d] ϵ^2/(2 d)}, {u, v}, {r, ϵ, rmax}][[1]]
```

```
Iter[a_, h_, p_, d_, rmax_, ϵ_, b_, η_, j_, Nmax_] :=
 Module[{M = H[a + h, p, d, rmax, ϵ]}, If[Or[Abs[b - M] < η, j > Nmax],
   {j, N[a], M, M - b, N[h], IGN[p, d, a, rmax, ϵ], p, K[Theta[p, d], p]},
   If[M < b, Iter[a + h, h, p, d, rmax, ϵ, M, η, j + 1, Nmax],
    Iter[a + h, -h / 2, p, d, rmax, ϵ, M, η, j + 1, Nmax]]]]
```

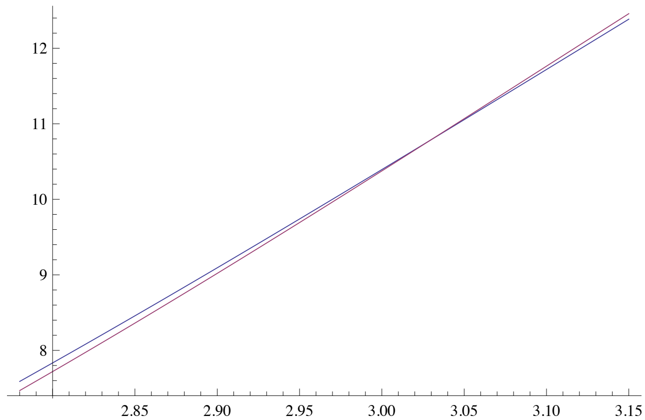
```
Init[a_, h_, p_, d_, rmax_, ϵ_, η_, Nmax_] :=
 Iter[a, h, p, d, rmax, ϵ, H[a, p, d, rmax, ϵ], η, 1, Nmax]
```

```
Nrm[p_, d_, a_, rmax_, ϵ_] := {z[rmax], w2[rmax], w[rmax]} /.
  NDSolve[{v'[r] + (d - 1) v[r]/r + a/Theta[p, d] Abs[u[r]]^(p-2) u[r] - (1 - Theta[p, d])/Theta[p, d] u[r] ==
       0, u'[r] == v[r], w'[r] == r^(d-1) Abs[u[r]]^p, w2'[r] == r^(d-1) Abs[u[r]]^2,
      z'[r] == r^(d-1) Abs[v[r]]^2, z[ϵ] == ((a + Theta[p, d] - 1)/Theta[p, d])^2 ϵ^(d+2)/(d^2 (d + 2)),
      w[ϵ] == ϵ^d/d, w2[ϵ] == ϵ^d/d, v[ϵ] == -(a + Theta[p, d] - 1)/Theta[p, d] ϵ/d,
```

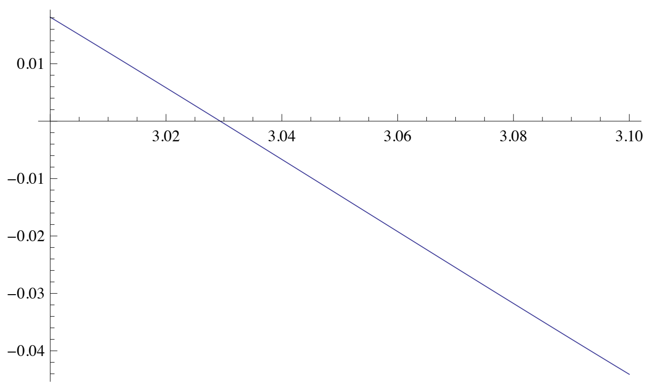
```
    u[ϵ] == 1 - (a + Theta[p, d] - 1)/Theta[p, d] ϵ^2/(2 d)}, {u, v, w, w2, z}, {r, ϵ, rmax}][[1]]

IGN[p_, d_, a_, rmax_, ϵ_] :=
 Module[{M = Nrm[p, d, a, rmax, ϵ]}, (M[[1]]^Theta[p, d] M[[2]]^(1-Theta[p, d]))/M[[3]]^(2/p)]

Fnagn[p_, d_, x_] := ac[d] - Sqrt[(x K[Theta[p, d], p])^(d/((d-1) Theta[p, d]))]

t[a_, p_, d_, rmax_, ϵ_] :=
 Evaluate[y2[rmax]/y0[rmax] /. NDSolve[{v'[r] + (d - 1) v[r]/r + a/Theta[p, d] Abs[u[r]]^(p-2) u[r] -
       (1 - Theta[p, d])/Theta[p, d] u[r] == 0, u'[r] == v[r],
     y0'[r] == r^(d-1) u[r], y2'[r] == r^(d+1) u[r], v[ϵ] == -(a + Theta[p, d] - 1)/Theta[p, d] ϵ/d,
     u[ϵ] == 1 - (a + Theta[p, d] - 1)/Theta[p, d] ϵ^2/(2 d), y0[ϵ] == ϵ^d/d - (a + Theta[p, d] - 1)/Theta[p, d] ϵ^(d+1)/(d (d + 1)),
     y2[ϵ] == ϵ^(d+2)/(d + 2) - (a + Theta[p, d] - 1)/Theta[p, d] ϵ^(d+3)/(d (d + 3))}, {u, v, y0, y2}, {r, ϵ, rmax}]][[1]]

VerificationGN[a_, h_, p_, d_, rmax_, ϵ_, η_, amin_, amax_, Nmax_] :=
 Module[{M = Init[a, h, p, d, rmax, ϵ, η, Nmax]},
  {Plot[H[aa, p, d, rmax, ϵ], {aa, amin, amax}, DisplayFunction → Visualize],
   {M, {Fnagn[p, d, M[[6]]], t[M[[6]], p, d, rmax, ϵ], Theta[p, d]}},
   Show[F[M[[2]], p, d, rmax, ϵ, Visualize, Automatic],
    DisplayFunction → Visualize] }]

ResultGN[a_, h_, p_, d_, rmax_, ϵ_, η_, amin_, amax_, Nmax_] :=
 Module[{M = Init[a, h, p, d, rmax, ϵ, η, Nmax]}, {M[[6]], t[M[[6]], p, d, rmax, ϵ]}}]
```

Computing p^*

$$\texttt{Plot}\left[\left\{\frac{1}{\texttt{Cstar[p, 5, LambdaFS[p, Theta[p, 5], 5], Theta[p, 5]]}},\right.\right.$$
$$\left.\left.\texttt{ResultGN}\left[1, 1, p, 5, 10, 10^{-12}, 10^{-15}, 1, 20, 100\right]\texttt{[[1]] S[5]}^{\frac{p-2}{p}}\right\}, \{p, 2.78, 3.15\}\right]$$



$$\texttt{Plot}\left[\frac{1}{\texttt{Cstar[p, 5, LambdaFS[p, Theta[p, 5], 5], Theta[p, 5]]}} -\right.$$
$$\left.\texttt{ResultGN}\left[1, 1, p, 5, 10, 10^{-12}, 10^{-15}, 1, 20, 100\right]\texttt{[[1]] S[5]}^{\frac{p-2}{p}}, \{p, 3.0, 3.1\}\right]$$



$$\texttt{FbarTheta[p\_] :=} \frac{1}{\texttt{Cstar[p, 5, LambdaFS[p, Theta[p, 5], 5], Theta[p, 5]]}} -$$
$$\texttt{ResultGN}\left[1, 1, p, 5, 10, 10^{-12}, 10^{-15}, 1, 20, 100\right]\texttt{[[1]] S[5]}^{\frac{p-2}{p}}$$

1.81597×10^{-7}

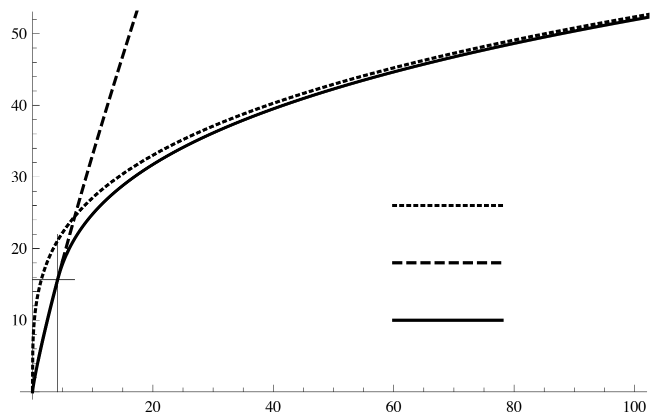
4.00811×10^{-7}

```
LinInterp[x1_, x2_, y1_, y2_] := x1 - y1 (x2 - x1)/(y2 - y1)
LinInterp[3.02, 3.03, FbarTheta[3.02], FbarTheta[3.03]]
3.02935

FbarTheta[3.02935]
FbarTheta[3.02936]
3.68746 × 10^-6
-2.27632 × 10^-6

Theta[3.02935, 5]
0.849481
```

Figure 1

```
Psym1a = Plot[1/Cstar[2.8, 5, λ, 1],
  {λ, 5, 52}, PlotStyle → {Thickness[0.005], Dashed, Black}];
Psym1b = Plot[1/Cstar[2.8, 5, λ, 1], {λ, 0, 5}, PlotStyle → {Thickness[0.005], Black}];

VerificationGN[1, 1, 2.8, 5, 10, 10^-12, 10^-15, 1, 20, 100];
KGNinv = ResultGN[1, 1, 2.8, 5, 10, 10^-12, 10^-15, 1, 20, 100][[1]]
3.03245

FitGN[p_, d_, θ_] := Plot[(θ^θ KGNinv S[d]^((p-2)/p))/(Theta[p, d]^Theta[p,d] (θ - Theta[p, d])^(θ-Theta[p,d])) λ^(θ-Theta[p,d]),
  {λ, 0, 130}, PlotStyle → {Dashing[Tiny], Thickness[0.005], Black}]
```

```
Legende[h_, PS_] := ListLinePlot[{{60, h}, {78, h}}, PlotStyle → PS]

Show[Psym1a, Psym1b, HH[22, μFS[5, 2.8]],
 LL[1 / Cstar[2.8, 5, μFS[5, 2.8], 1], 7], PlotTable[L811a, 3, 4],
 PlotTable[L811b, 3, 4], FitGN[2.8, 5, 1], Legende[10, {Thickness[0.005], Black}],
 Legende[18, {Thickness[0.005], Dashed, Black}],
 Legende[26, {Dashing[Tiny], Thickness[0.005], Black}],
 PlotRange → {{0, 100}, {0, 52}}, AxesOrigin → {0, 0}]
```




```
LambdaTheta[p_, d_, θ_] := (θ - (1 - θ) (p - 2)/(p + 2)) μFS[d, p]

LambdaTheta[2.8, 5, 0.8]

3.19444
```

Figure 2

```
Psym2a = Plot[1/Cstar[2.8, 5, λ, 0.8], {λ, LambdaTheta[2.8, 5, 0.8], 52},
   PlotStyle → {Thickness[0.005], Dashed, Black}]; Psym2b =
  Plot[1/Cstar[2.8, 5, λ, 0.8], {λ, 0, 3.5}, PlotStyle → {Thickness[0.005], Black}];

Legende[h_, PS_] := ListLinePlot[{{30, h}, {40, h}}, PlotStyle → PS]

Show[Psym2a, Psym2b, HH[10, LambdaTheta[2.8, 5, 0.8]],
 LL[1 / Cstar[2.8, 5, LambdaTheta[2.8, 5, 0.8], 0.8], 7],
 PlotTableEnergyTheta[2.8, 0.8, L811b],
 FitGN[2.8, 5, 0.8], Legende[9, {Thickness[0.005], Black}],
 Legende[10, {Thickness[0.005], Dashed, Black}],
 Legende[11, {Dashing[Tiny], Thickness[0.005], Black}],
 PlotRange → {{0, 50}, {8, 15}}, AxesOrigin → {0, 8}]
```

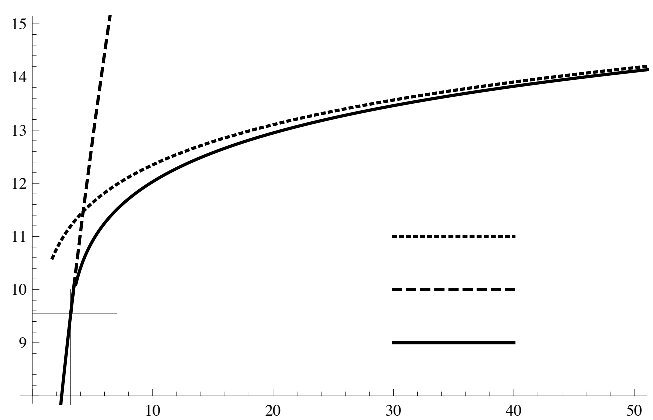


Figure 3

```
Psym3a = Plot[1/Cstar[2.8, 5, λ, 0.72], {λ, LambdaTheta[2.8, 5, 0.72], 52},
   PlotStyle → {Thickness[0.005], Dashed, Black}];
Psym3b = Plot[1/Cstar[2.8, 5, λ, 0.72], {λ, 0, LambdaTheta[2.8, 5, 0.72]},
   PlotStyle → {Thickness[0.005], Black}];

Legende[h_, PS_] := ListLinePlot[{{5, h}, {6, h}}, PlotStyle → PS]

Show[Psym3a, Psym3b, HH[8.2, LambdaTheta[2.8, 5, 0.72]],
 LL[1 / Cstar[2.8, 5, LambdaTheta[2.8, 5, 0.72], 0.72], 3.5],
 PlotTableEnergyTheta[2.8, 0.72, L811b], FitGN[2.8, 5, 0.72],
 Legende[8.4, {Thickness[0.005], Black}],
 Legende[8.7, {Thickness[0.005], Dashed, Black}],
 Legende[9, {Dashing[Tiny], Thickness[0.005], Black}],
 PlotRange → {{2, 8}, {7, 9}}, AxesOrigin → {2, 7}]
```

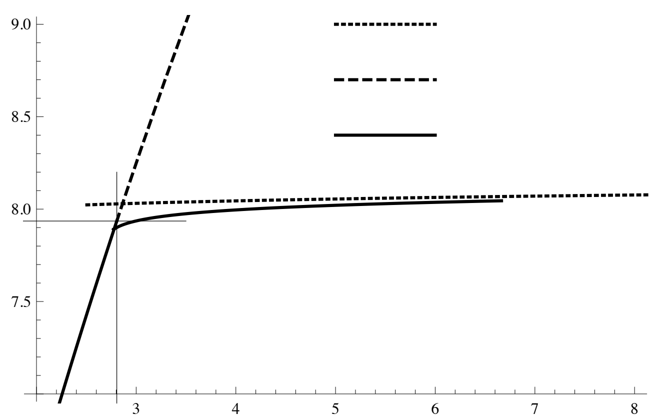


Figure (additional)

```
P280new80 = PlotTableEnergyThetabis[2.8, 0.8, LL280err12moinspremierterme];
P280star80a = Plot[1/Cstar[2.8, 5, λ, 0.8], {λ, 0, LambdaTheta[2.8, 5, 0.8]},
   PlotRange → {{3, 3.8}, {9, 10.5}}, PlotStyle → {Thickness[0.01], Black}];
P280star80b = Plot[1/Cstar[2.8, 5, λ, 0.8], {λ, LambdaTheta[2.8, 5, 0.8], 4},
   PlotRange → {{3, 3.8}, {9, 10.5}},
   PlotStyle → {Thickness[0.005], Opacity[0.6], Dashed, Black}];

Show[P280star80a, P280star80b, P280new80, HH[9.7, LambdaTheta[2.8, 5, 0.8]],
 LL[1 / Cstar[2.8, 5, LambdaTheta[2.8, 5, 0.8], 0.8], 3.3],
 AspectRatio → 1, AxesOrigin → {3, 9}]
```

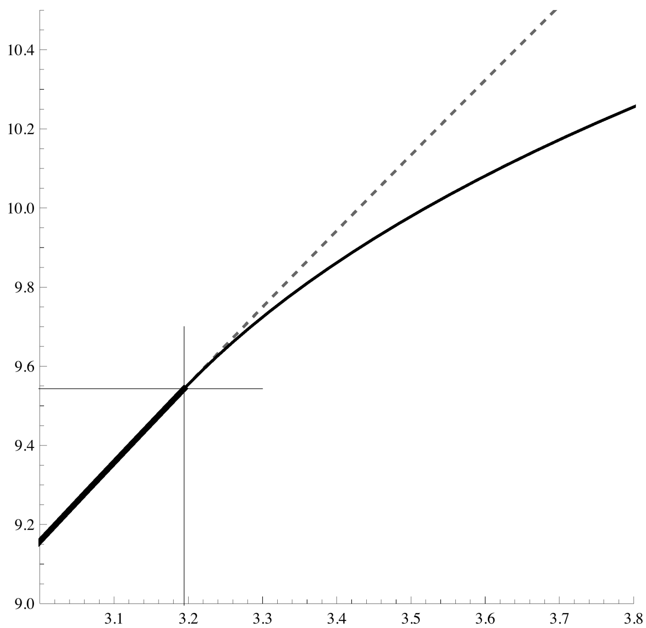


Figure 4

```
P280new95 = PlotTableEnergyThetabis[2.8, 0.95, LL280err12moinspremierterme];
P280star95a = Plot[1/Cstar[2.8, 5, λ, 0.95], {λ, 0, LambdaTheta[2.8, 5, 0.95]},
   PlotRange → All, PlotStyle → {Thickness[0.01], Black}];
P280star95b = Plot[1/Cstar[2.8, 5, λ, 0.95], {λ, LambdaTheta[2.8, 5, 0.95], 10},
   PlotRange → All, PlotStyle → {Thickness[0.005], Opacity[0.6], Dashed, Black},
   PlotRange → {{1, 10}, {5, 20}}];

Show[P280star95a, P280star95b, P280new95, HH[16, LambdaTheta[2.8, 5, 0.95]],
 LL[1 / Cstar[2.8, 5, LambdaTheta[2.8, 5, 0.95], 0.95], 5], AspectRatio → 1,
 AxesOrigin → {2.5, 10}, PlotRange → {{2.5, 7.5}, {10, 19}}]
```

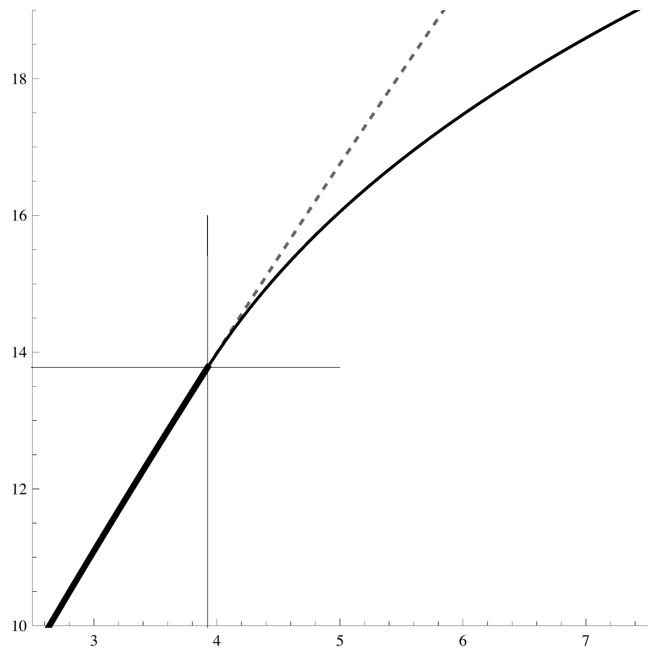


Figure 5

```
P280new72 = PlotTableEnergyThetabis[2.8, 0.72, LL280err12moinspremierterme];
P280star72b = Plot[1/Cstar[2.8, 5, λ, 0.72], {λ, LambdaTheta[2.8, 5, 0.72], 4},
   PlotStyle → {Thickness[0.005], Opacity[0.6], Dashed, Black}];
P280star72a = Plot[1/Cstar[2.8, 5, λ, 0.72], {λ, 0, LambdaTheta[2.8, 5, 0.72]},
   PlotStyle → {Thickness[0.01], Black}];
```

```
Show[P280star72a, P280star72b, P280new72, HH[9.7, LambdaTheta[2.8, 5, 0.72]],
 LL[1 / Cstar[2.8, 5, LambdaTheta[2.8, 5, 0.72], 0.72], 3.3],
 PlotRange → {{2.76, 2.82}, {7.87, 7.95}}, AspectRatio → 1, AxesOrigin → {2.76, 7.87}]
```

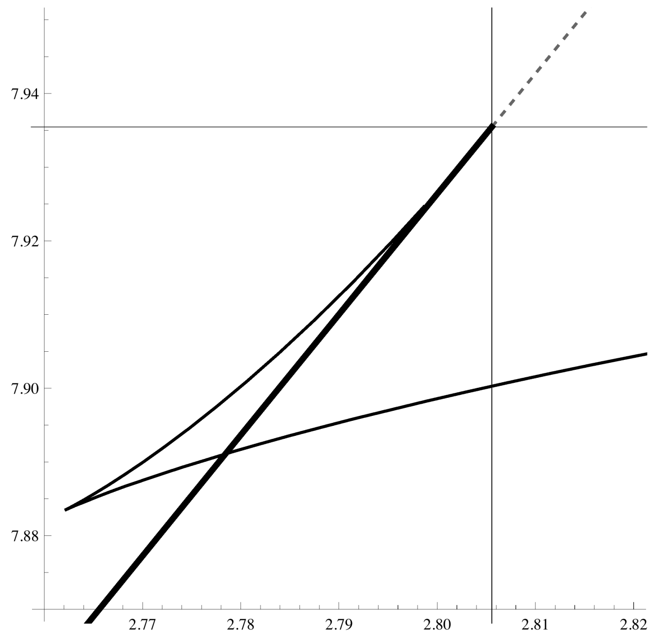
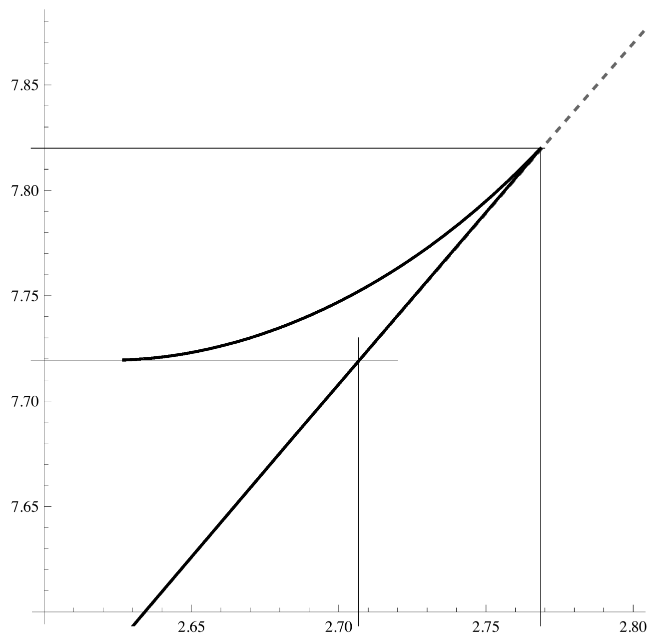


Figure 6

```
P280new = PlotTableEnergyThetabis[2.8, Theta[2.8, 5], LL280err12moinspremierterme];
P280star = Plot[1/Cstar[2.8, 5, λ, Theta[2.8, 5]], {λ, 0, 4},
   PlotStyle → {Thickness[0.005], Opacity[0.6], Dashed, Black}];

Show[P280star, P280new, HH [7.73, 2.7067], HH [7.82, 2.7685],
 LL[ResultGN[1, 1, 2.8, 5, 10, 10^-12, 10^-15, 1, 20, 100][[1]] S[5]^((p-2)/p) /. p → 2.8, 2.72],
 LL[7.82, 2.77], PlotRange → {{2.6, 2.8}, {7.6, 7.88}},
 AspectRatio → 1, AxesOrigin → {2.6, 7.6}]
```



Figures 7

```
LL315 = {};
For[i = 0, i < Length[LL315errmoins10], i++,
 If[LL315errmoins10[[i]][[3]] < LambdaFS[3.15, 1, 5],
  Print[No, ":", i, ":", LL315errmoins10[[i]][[3]]],
  LL315 = Append[LL315, LL315errmoins10[[i]]]]]
```

```
No:2:2.65951

No:3:2.66392

No:4:2.66832

No:5:2.67273

No:6:2.67713

No:7:2.68152

No:8:2.68594

No:9:2.69035

No:10:2.69477

No:11:2.69918
```

```
CrossPlot[θ_, s_, e_, n_, w_] :=
 Module[{PointFS = {LambdaFS[3.15, θ, 5], 1/Cstar[3.15, 5, LambdaFS[3.15, θ, 5], θ]}},
  Show[{ListLinePlot[{{PointFS[[1]] - w, PointFS[[2]]},
      {PointFS[[1]] + e, PointFS[[2]]}}, PlotStyle → {Thickness[0.001], Black}],
    ListLinePlot[{{PointFS[[1]], PointFS[[2]] - s},
      {PointFS[[1]], PointFS[[2]] + n}}, PlotStyle → {Thickness[0.001], Black}]}]]
```

```
Fig7[θ_, λR_, AO_, PR_, kpass_] :=
 Module[{TT315d5315errmoins10 = Table[{Lambdabis[θ, LL315, n],
      Energybis[3.15, θ, LL315, n], n}, {n, 1, Length[LL315]}]},
  Show[{ListLinePlot[Table[{Abs[TT315d5315errmoins10[[n]][[1]]],
       TT315d5315errmoins10[[n]][[2]]}, {n, 5, Length[TT315d5315errmoins10]}],
     PlotStyle → {Thickness[0.005], Black}, PlotRange → All],
    Plot[1/Cstar[3.15, 5, λ, θ], {λ, λR[[1]], LambdaFS[3.15, θ, 5]},
     PlotStyle → {Thickness[0.005], Black}],
    Plot[1/Cstar[3.15, 5, λ, θ], {λ, LambdaFS[3.15, θ, 5], λR[[2]]},
     PlotStyle → {Thickness[0.005], Opacity[0.6], Dashed, Black}],
    CrossPlot[θ, kpass[[1]], kpass[[2]], kpass[[3]], kpass[[4]]]},
   PlotRange → PR, AxesOrigin → AO]]
```

Figure 7.1

```
Fig7[1, {0, 10}, {0, 10}, {{0, 20}, {10, 22}}, {15, 1, 2, 3}]
```

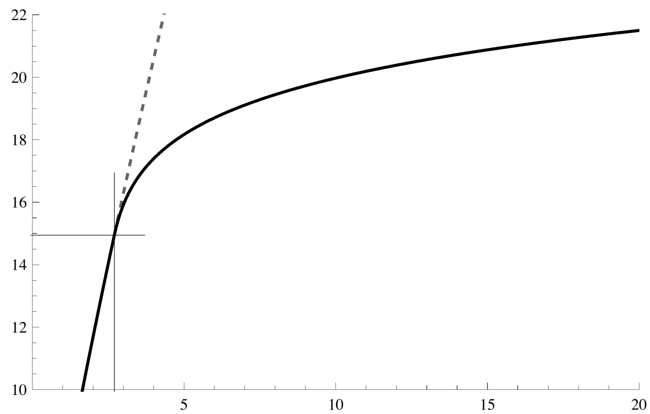


Figure 7.2

```
Fig7[0.95, {0, 10}, {0, 10}, {{0, 20}, {10, 18}}, {15, 1, 1, 3}]
```

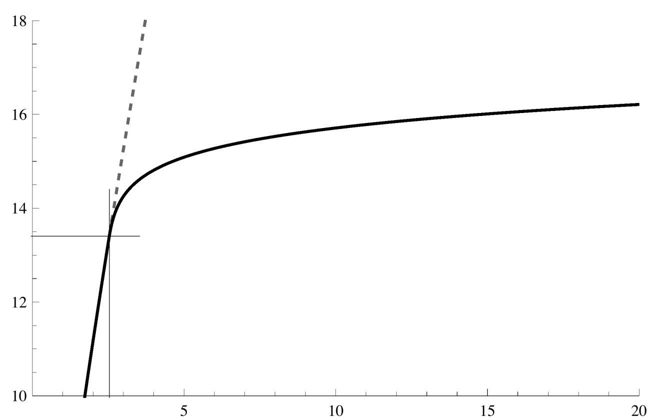
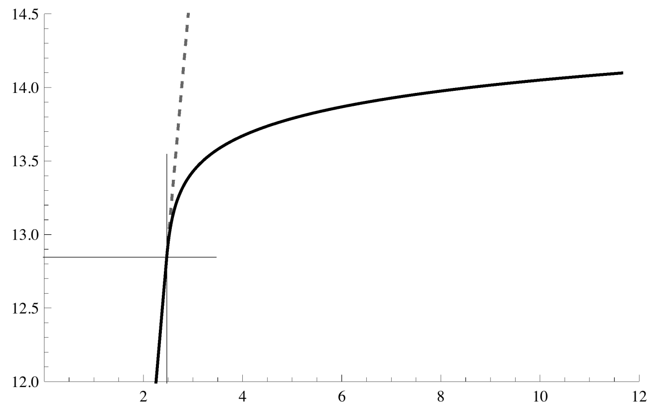


Figure 7: additional

```
Fig7[0.93, {0, 10}, {0, 12}, {{0, 12}, {12, 14.5}}, {15, 1, 0.7, 3}]
```




```
Fig7[0.915, {0, 10}, {2, 12}, {{2, 3.7}, {12, 12.8}}, {15, 0.25, 0.2, 1}]
```

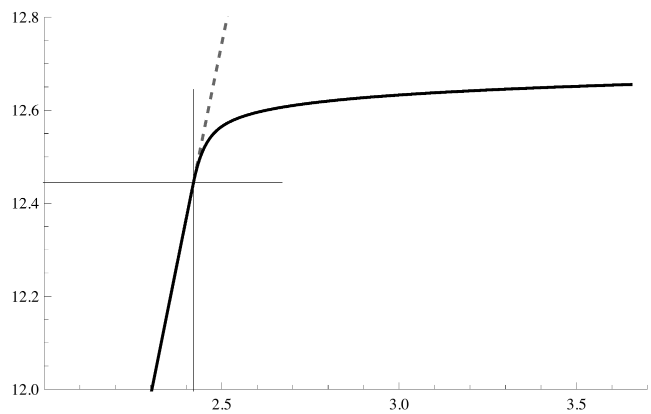
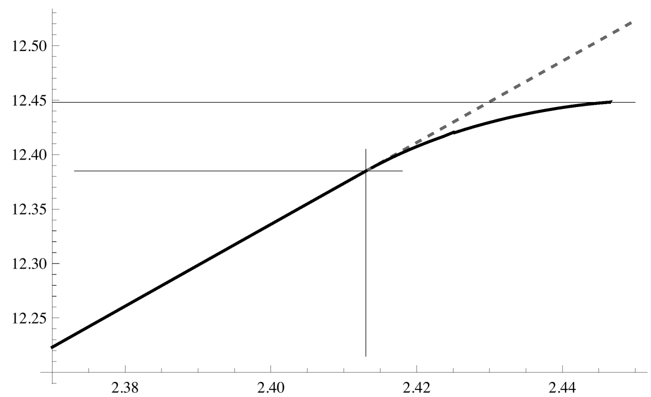


Figure 7.3

```
ResultGN[1, 1, 3.15, 5, 10, 10^-12, 10^-15, 1, 20, 100][[1]] S[5]^((p-2)/p) /. p → 3.15
AsympGN[p_, d_, l_, r_] :=
 Module[{LevelGN = ResultGN[1, 1, 3.15, 5, 10, 10^-12, 10^-15, 1, 20, 100][[1]] S[d]^((p-2)/p)},
  ListLinePlot[{{l, LevelGN}, {r, LevelGN}}, PlotStyle → {Thickness[0.001], Black}]]
12.4579
```

```
Fig7bis[θ_, λR_, AO_, PR_, kpass_, stop_] := Module[{TT315d5315errmoins10 =
    Table[{Lambdabis[θ, LL315, n], Energybis[3.15, θ, LL315, n], n}, {n, 1, stop}]},
  Show[{ListLinePlot[Table[{Abs[TT315d5315errmoins10[[n]][[1]]],
        TT315d5315errmoins10[[n]][[2]]}, {n, 5, Length[TT315d5315errmoins10]}],
      PlotStyle → {Thickness[0.005], Black}, PlotRange → All],
    Plot[1/Cstar[3.15, 5, λ, θ], {λ, λR[[1]], LambdaFS[3.15, θ, 5]},
      PlotStyle → {Thickness[0.005], Black}],
    Plot[1/Cstar[3.15, 5, λ, θ], {λ, LambdaFS[3.15, θ, 5], λR[[2]]},
      PlotStyle → {Thickness[0.005], Opacity[0.6], Dashed, Black}],
    CrossPlot[θ, kpass[[1]], kpass[[2]], kpass[[3]], kpass[[4]]]},
   PlotRange → PR, AxesOrigin → AO]]

AsympGN[p_, d_, l_, r_] := Module[{LevelGN = 12.448},
  ListLinePlot[{{l, LevelGN}, {r, LevelGN}}, PlotStyle → {Thickness[0.001], Black}]]

Show[Fig7bis[Theta[3.15, 5], {2.37, 2.45}, {2.37, 12.2},
  All, {0.17, 0.005, 0.02, 0.04}, 67], AsympGN[3.15, 5, 2.37, 2.45]]
```



3 Other plots, asymptotic regime and expansion at the bifurcation point: Mathematica

Computations at the bifurcation point

Definitions

$$f[q\_] := \sqrt{\pi}\,\frac{\text{Gamma}\left[\frac{q}{2}\right]}{\text{Gamma}\left[\frac{q+1}{2}\right]};$$

$$\text{I2}[p\_] := f\left[\frac{4}{p-2}\right]$$

$$\text{Ip}[p\_] := \frac{4}{p+2}\,\text{I2}[p]$$

$$\text{J2}[p\_] := \frac{4}{(p+2)\,(p-2)}\,\text{I2}[p]$$

$$\varkappa d = \sqrt{2}\,\sqrt{\frac{d-1}{d+2}};$$

$$\mu\text{FS}[p\_] := 4\,\frac{d-1}{p^2-4}$$

$$\text{bmu}[p\_] := \frac{4\,(d-1)^2\,(p-1)^2\,p^3\,\left(2\,p\,(5\,p-6)\,-d\,\left(12-16\,p+p^2\right)\right)}{(2+d)\,(2-3\,p)^2\,(-2+p)\,(2+p)^2\,(-6+5\,p)}$$

$$\alpha[p\_,\,\mu\_] := \left(\frac{p\,\mu}{2}\right)^{\frac{1}{p-2}}$$

$$\beta[p\_,\,\mu\_] := \frac{p-2}{2}\,\sqrt{\mu}$$

$$\text{A0}[p\_,\,\mu\_] := \frac{\alpha[p,\,\mu]^{p-1}}{2\,\beta[p,\,\mu]^2}\,\frac{p^2\,(p-1)\,(p-2)}{3\,p-2}\,\mu$$

$$\text{B0}[p\_,\,\mu\_] := \frac{\alpha[p,\,\mu]^{2\,p-3}}{2\,\beta[p,\,\mu]^2}\,(p-1)\,(p-2)$$

$$\text{B2}[p\_,\,\mu\_] := \varkappa d\;\text{B0}[p,\,\mu]$$

Solving the equation for $\chi_{2,2p-3}$

```
smax2[p_] := 5/6 p + 1/4

Tir2[p_, a_] :=
 N[χ[smax2[p]] /. NDSolve[{χ''[s] == -Cosh[s]^(-2 (2p-3)/(p-2)) + 2/(p - 2)^2 (2 - p (p - 1) Sech[s]^2) χ[s] +
       (2 d)/(d - 1) (p + 2)/(p - 2) χ[s], χ[0] == a, χ'[0] == 0}, {χ, χ'}, {s, 0, smax2[p]}]][[1]]
F2[p_, a_, PS_] := Plot[χ[r] /. NDSolve[{χ''[s] == -Cosh[s]^(-2 (2p-3)/(p-2)) +
       2/(p - 2)^2 (2 - p (p - 1) Sech[s]^2) χ[s] + (2 d)/(d - 1) (p + 2)/(p - 2) χ[s],
     χ[0] == a, χ'[0] == 0}, {χ, χ'}, {s, 0, smax2[p]}],
  {r, 0, smax2[p]}, PlotRange → All, DisplayFunction → PS]

Giter2[p_, a_, h_, b_, n_] :=
 If[Abs[h] < eps || n > 200, N[{a, h, n}], Module[{M = Tir2[p, a]},
   If[M * b > 0, Giter2[p, a + h, h, M, n + 1], Giter2[p, a, -h / 2, M, n + 1]]]]
SearchG2[p_] := Giter2[p, 0, 0.1, Tir2[p, 0], 1]

FP2[p_, PS_] := Module[{a = SearchG2[p][[1]]}, F2[p, a, PS]]

Coef2[p_, a_] := N[{2 c[smax2[p]], 2 ccc[smax2[p]], 2 cccc[smax2[p]]} /. NDSolve[
    {χ''[s] == -Cosh[s]^(-2 (2p-3)/(p-2)) + 2/(p - 2)^2 (2 - p (p - 1) Sech[s]^2) χ[s] + (2 d)/(d - 1) (p + 2)/(p - 2) χ[s],
     c'[s] == χ[s] Cosh[s]^(-2 (2p-3)/(p-2)), ccc'[s] == χ[s]^2, cccc'[s] == χ'[s]^2,
     χ[0] == a, χ'[0] == 0, c[0] == 0, ccc[0] == 0, cccc[0] == 0},
    {χ, χ', c, ccc, cccc}, {s, 0, smax2[p]}]][[1]]

CoefA2[p_] := Module[{a = SearchG2[p][[1]]}, Coef2[p, a]]

CFb2[p_] := CoefA2[p][[1]]
CFe2[p_] := CoefA2[p][[2]]
CFf2[p_] := CoefA2[p][[3]]
```

The order ϵ^4 term

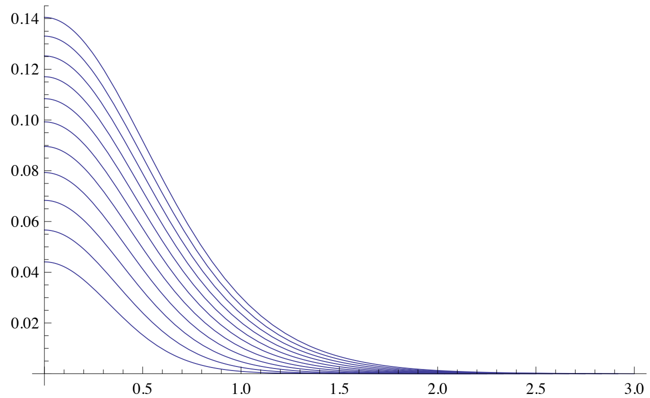
```
Lalg2[p_, y_] := 4 (-1 + d)^2 (-1 + p) p^3 / (p + 2)^2 ((-2 + p) p / ((2 - 3 p)^2 (-6 + 5 p)) + 2 (d - 1)/(d + 2) (p - 1)/(p - 2)^2 y)
L[p_] := Lalg2[p, CFb2[p] / Ip[p]]
U0[p_] := 2 p (p - 1) / (p - 2)^2
λ1[p_] := -(U0[p] + 1/2 - 1/2 Sqrt[4 U0[p] + 1])
Lapprox[p_] :=
 Simplify[Lalg2[p, 16 p (p - 1) (3 p - 4) / ((3 p - 2) (5 p - 6) (7 p - 10)) 1/(λ1[p] + 4/(p-2)^2 (1 + d (p-2) (p+2)/(2 (d-1))))] /.
   Sqrt[(2 - 3 p)^2 / (-2 + p)^2] → (3 p - 2)/(p - 2)]
Lapprox[
 p]
```

$$\left(4\,(-1+d)^2\,(-1+p)\,p^4\left(\frac{(-2+p)^3}{(2-3p)^2}+\left(64\,(-1+d)\,(-1+p)^2\,(-4+3p)\right)\Big/\left((2+d)\,(-2+3p)\,(-10+7p)\left(-1+\sqrt{\frac{(2-3p)^2}{(-2+p)^2}}-\frac{4\,(-1+p)\,p}{(-2+p)^2}+\frac{-8+4d\left(-2+p^2\right)}{(-1+d)\,(-2+p)^2}\right)\right)\right)\right)\Big/\left((-6+5p)\left(-4+p^2\right)^2\right)$$

Numerical computations for d = 5

```
eps = 10^-12;
d = 5;
```

```
Show[Table[FP2[p, Identity], {p, 2.3, 3.3, 0.1}],
 DisplayFunction -> $DisplayFunction]
```



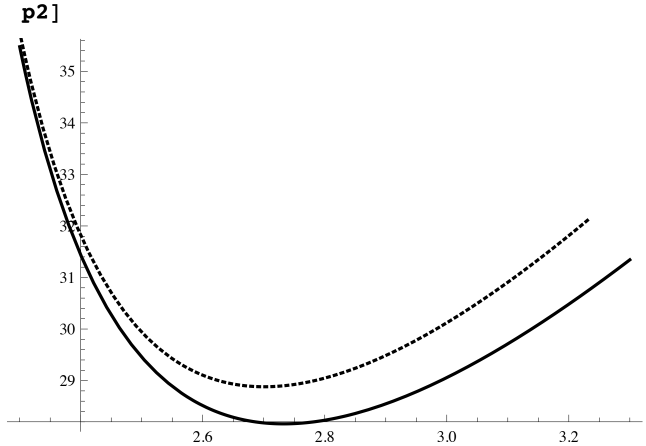

```
Simplify[Lapprox[p] - 2 bmu[p] /. d → 5]
pmax = p /. FindRoot[-3600 + 2816 p + 3512 p^2 - 3664 p^3 + 747 p^4 == 0, {p, 2, 10 / 3}]
```

$$\frac{64\ (-1+p)\ p^3\ \left(-3600+2816\ p+3512\ p^2-3664\ p^3+747\ p^4\right)}{21\ (-2+p)\ (2+p)^3\ (-2+3\ p)\ (-6+5\ p)\ (-10+7\ p)}$$

3.23233

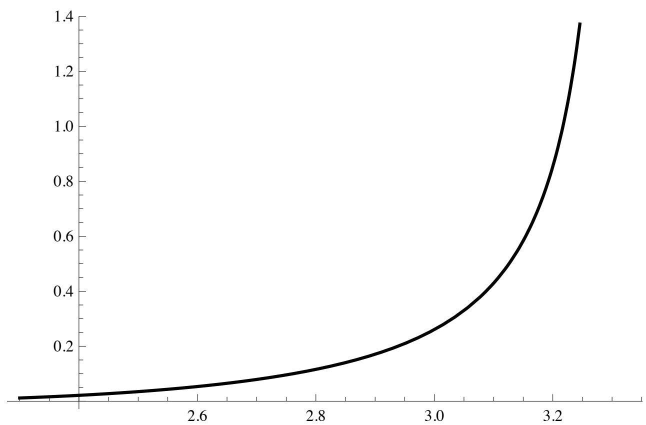
Plot of the order ϵ^4 term

```
p1 = Plot[L[p], {p, 2.3, 3.3}, PlotStyle → {Thickness[0.005], Black}];
p2 = Plot[Lapprox[p], {p, 2.3, pmax},
   PlotStyle → {Thickness[0.005], Dashing[Tiny], Black}];
Show[
 p1,
 p2]
```

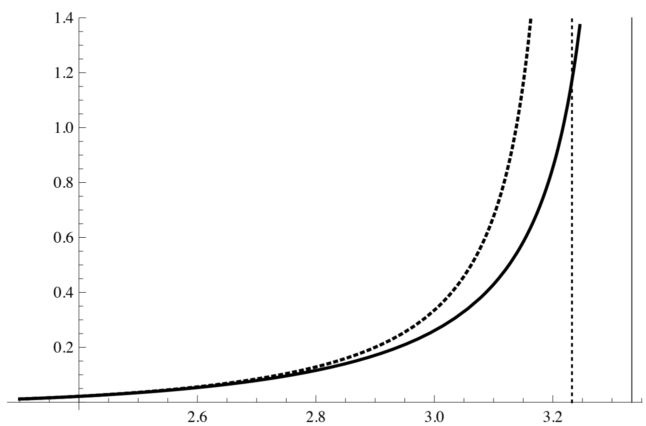


Plot o f the $c_{p,d}$ constant

```
cpd[p_] := (p^2 - 4)/(4 (2 bmu[p] - L[p]))

cpdapprox[p_] := (p^2 - 4)/(4 (2 bmu[p] - Lapprox[p]))

p3 = Plot[cpd[p], {p, 2.3, 3.33}, PlotStyle → {Thickness[0.005], Black}]
```



```
p4 = Plot[If[cpdapprox[p] < 1.4, cpdapprox[p], {}], {p, 2.3, pmax},
   PlotStyle → {Thickness[0.005], Dashing[Tiny], Black}, PlotRange → All];
Show[p3, p4, ListLinePlot[{{pmax, 0}, {pmax, 1.4}},
  PlotStyle → {Thickness[0.003], Dashing[Tiny], Black}],
 ListLinePlot[{{10/3, 0}, {10/3, 1.4}}, PlotStyle → Black]]
```



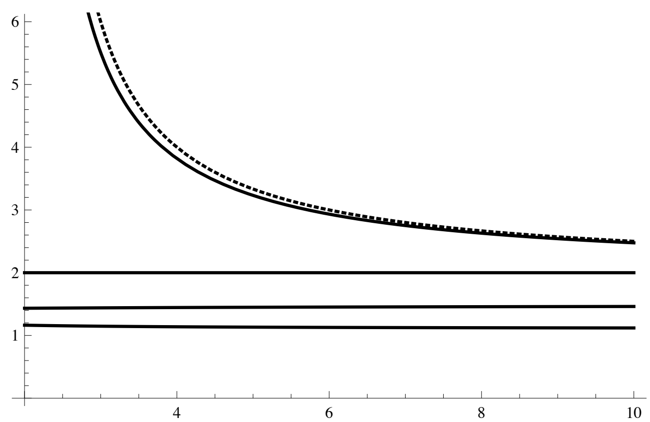
Consistency : an estimate

```
Simplify[2 bmu[p] - Lapprox[p] /. Sqrt[(2 - 3 p)^2/(-2 + p)^2] → (3 p - 2)/(p - 2)]
Limit[(p - 2) %, p → 2]
h[d_] := p /. Solve[18 (-2 + p)^2 p (-4 + 3 p) + d (-240 + 384 p - 536 p^2 + 592 p^3 - 227 p^4) +
      d^2 (-240 + 160 p + 368 p^2 - 400 p^3 + 103 p^4) == 0, p]
```

$$-\left(4\ (-1+d)^2\ (-1+p)\ p^3\ \left(18\ (-2+p)^2\ p\ (-4+3\ p)+d\ \left(-240+384\ p-536\ p^2+592\ p^3-227\ p^4\right)+d^2\ \left(-240+160\ p+368\ p^2-400\ p^3+103\ p^4\right)\right)\right)\Big/\left((1+d)\ (2+d)\ (-2+p)\ (2+p)^3\ (-2+3\ p)\ (-6+5\ p)\ (-10+7\ p)\right)$$

$$\frac{4\ (-1+d)^2\ d}{(1+d)\ (2+d)}$$

```
Show[Plot[h[d], {d, 2, 10}, PlotStyle → {Thickness[0.005], Black}],
 Plot[2, {d, 2, 10}, PlotStyle → {Thickness[0.005], Black}],
 Plot[2 d/(d - 2), {d, 2, 10}, PlotStyle → {Thickness[0.005], Dashing[Tiny], Black}],
 PlotRange → {All, {0, 6}}, AxesOrigin → {2, 0}]
Chop[N[h[5]]]
```

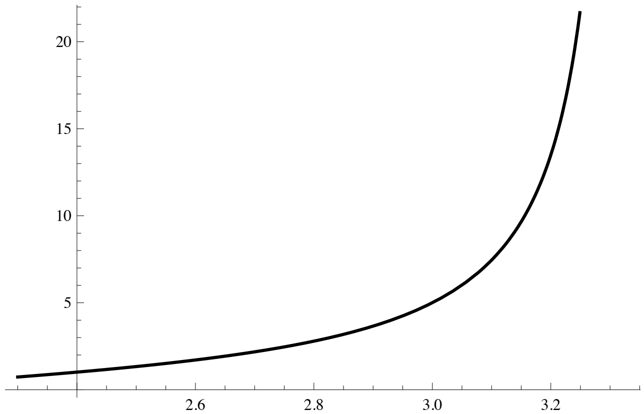


{1.44913, 3.23233, -0.908721, 1.13221}

Plot of the function τ '

```
τprime[p_] := (p - 2)/(p + 2) + 16 (d - 1)^2 p^2/((p - 2) (2 + p)^3) cpd[p]
```

```
Plot[τprime[p], {p, 2.3, 3.33}, PlotStyle → {Thickness[0.005], Black}]
```



Plot of the critical value of the parameter θ

$$S[d\_] := \frac{2\,\pi^{\frac{d}{2}}}{\text{Gamma}\left[\frac{d}{2}\right]}$$

Cstar[p_, d_, λ_, θ_] :=

$$S[d]^{-\frac{p-2}{p}} \left(\frac{(p-2)^2}{2+(2\,\theta-1)\,p}\right)^{\frac{p-2}{2\,p}} \left(\frac{2+(2\,\theta-1)\,p}{2\,p\,\theta}\right)^{\theta} \left(\frac{4}{p+2}\right)^{\frac{6-p}{2\,p}} \left(\frac{\text{Gamma}\left[\frac{2}{p-2}+\frac{1}{2}\right]}{\sqrt{\pi}\ \text{Gamma}\left[\frac{2}{p-2}\right]}\right)^{\frac{p-2}{p}} \lambda^{\frac{p-2}{2\,p}-\theta}$$

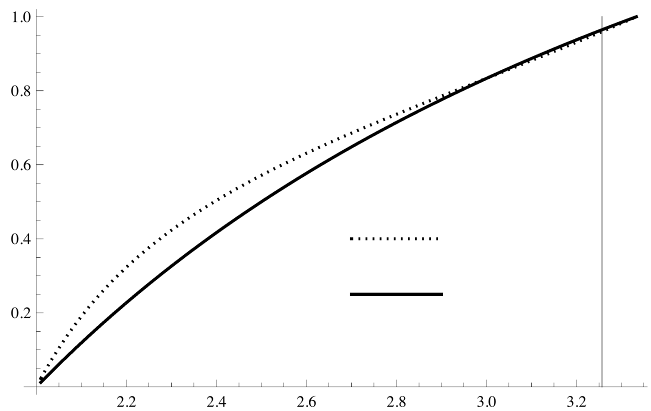
$$\text{LambdaTheta}[q\_, \theta\_] := \left(\theta-(1-\theta)\,\frac{q-2}{q+2}\right)\,\mu\text{FS}[p]\ /.\ p \to q$$

$$\text{Theta2}[p\_] := \frac{\tau\text{prime}[p]}{1+\tau\text{prime}[p]}$$

$$\text{Theta}[p\_] := d\,\frac{p-2}{2\,p}$$

```
PTheta = Show[Plot[Theta2[p], {p, 2.01, 3.333},
    PlotStyle → {Dashing[Tiny], Thickness[0.005], Black}],
   Plot[Theta[p], {p, 2.01, 3.333}, PlotStyle → {Thickness[0.005], Black}]];
```

```
Show[PTheta, ParametricPlot[{3.256494, x}, {x, 0, 1},
  PlotStyle → {Black, Opacity[0.6]}], ListLinePlot[{{2.7, 0.4}, {2.9, 0.4}},
  PlotStyle → {Dashing[Tiny], Thickness[0.005], Black}],
 ListLinePlot[{{2.7, 0.25}, {2.9, 0.25}}, PlotStyle → {Thickness[0.005], Black}],
 PlotRange → {{2, 3.33}, {0, 1}}, AxesOrigin → {2, 0}]
```

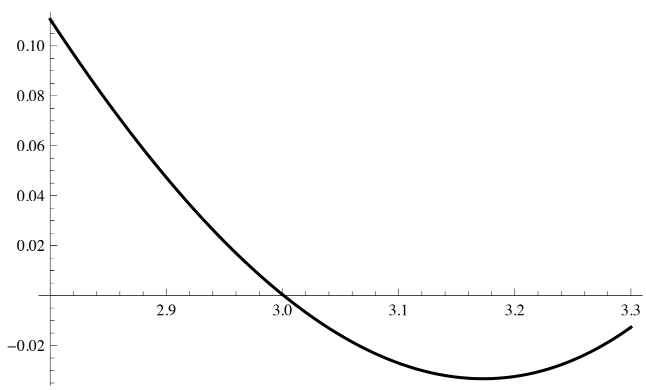


Plot of the differences and comparison with the criterion based on Gagliardo – Nirenberg inequalities

```
FTheta[p_] := Theta2[p] - Theta[p]
```

$$\texttt{LinInterp[x1\_, x2\_, y1\_, y2\_] := x1 - y1}\ \frac{\texttt{x2 - x1}}{\texttt{y2 - y1}}$$

```
Delta1 = ListLinePlot[Table[{p, 5 FTheta[p]}, {p, 2.8, 3.3, 0.001}],
  PlotStyle → {Thickness[0.005], Black}]
```



```
LinInterp[3.001, 3.002, FTheta[3.001], FTheta[3.002]]
FTheta[3.001115]
FTheta[3.00112]
Theta[3.001115]
Theta[3.00112]
```

3.00111

-3.48538×10^{-7}

-7.22749×10^{-7}

0.833953

0.833955

Plot of the differences and comparison with the criterion based on Gagliardo – Nirenberg inequalities

$$\text{ThetaGen[p\_, d\_] := d}\,\frac{\text{p - 2}}{\text{2 p}}$$

$$\mu\text{FSGen[p\_, d\_] := 4}\,\frac{\text{d - 1}}{\text{p}^2\text{ - 4}}$$

$$\text{Lambdastar1[p\_, }\theta\text{\_, mu\_] := }\theta\text{ mu - (1 - }\theta\text{) mu}\,\frac{\text{p - 2}}{\text{p + 2}}$$

$$\text{LambdaFS[p\_, }\theta\text{\_, d\_] := Lambdastar1[p, }\theta\text{, }\mu\text{FSGen[p, d]]}$$

$$\text{H[a\_, p\_, d\_, rmax\_, }\epsilon\text{\_] := Log[1 + u[rmax]\^{}2 + v[rmax]\^{}2] /.}$$
$$\text{NDSolve}\Big[\Big\{\text{v'[r] + (d - 1)}\,\frac{\text{v[r]}}{\text{r}} + \frac{\text{a}}{\text{ThetaGen[p, d]}}\,\text{Abs[u[r]]}^{\text{p}-2}\,\text{u[r]} - \frac{\text{1 - ThetaGen[p, d]}}{\text{ThetaGen[p, d]}}\,\text{u[r]} == 0,\ \text{u'[r]} == \text{v[r]},\ \text{v}[\epsilon] == -\frac{\text{a + ThetaGen[p, d] - 1}}{\text{ThetaGen[p, d]}}\,\frac{\epsilon}{\text{d}},$$
$$\text{u}[\epsilon] == 1 - \frac{\text{a + ThetaGen[p, d] - 1}}{\text{ThetaGen[p, d]}}\,\frac{\epsilon^2}{\text{2 d}}\Big\},\ \{\text{u, v}\},\ \{\text{r}, \epsilon, \text{rmax}\}\Big]\text{[[1]]}$$

$$\text{Nrm[p\_, d\_, a\_, rmax\_, }\epsilon\text{\_] :=}$$
$$\text{\{z[rmax], w2[rmax], w[rmax]\} /. NDSolve}\Big[\Big\{\text{v'[r] + (d - 1)}\,\frac{\text{v[r]}}{\text{r}} + \frac{\text{a}}{\text{ThetaGen[p, d]}}\,\text{Abs[u[r]]}^{\text{p}-2}\,\text{u[r]} - \frac{\text{1 - ThetaGen[p, d]}}{\text{ThetaGen[p, d]}}\,\text{u[r]} == 0,\ \text{u'[r]} == \text{v[r]},$$
$$\text{w'[r]} == \text{r}^{\text{d}-1}\,\text{Abs[u[r]]}^{\text{p}},\ \text{w2'[r]} == \text{r}^{\text{d}-1}\,\text{Abs[u[r]]}^2,\ \text{z'[r]} == \text{r}^{\text{d}-1}\,\text{Abs[v[r]]}^2,$$
$$\text{z}[\epsilon] == \left(\frac{\text{a + ThetaGen[p, d] - 1}}{\text{ThetaGen[p, d]}}\right)^2 \frac{\epsilon^{\text{d}+2}}{\text{d}^2\text{ (d + 2)}},\ \text{w}[\epsilon] == \frac{\epsilon^{\text{d}}}{\text{d}},\ \text{w2}[\epsilon] == \frac{\epsilon^{\text{d}}}{\text{d}},$$
$$\text{v}[\epsilon] == -\frac{\text{a + ThetaGen[p, d] - 1}}{\text{ThetaGen[p, d]}}\,\frac{\epsilon}{\text{d}},\ \text{u}[\epsilon] == 1 - \frac{\text{a + ThetaGen[p, d] - 1}}{\text{ThetaGen[p, d]}}\,\frac{\epsilon^2}{\text{2 d}}\Big\},$$

```
      {u, v, w, w2, z}, {r, ε, rmax}] [[1]]

IGN[p_, d_, a_, rmax_, ε_] :=
 Module[{M = Nrm[p, d, a, rmax, ε]}, (M[[1]]^ThetaGen[p, d] M[[2]]^(1-ThetaGen[p, d])) / M[[3]]^(2/p)]

t[a_, p_, d_, rmax_, ε_] :=
 Evaluate[y2[rmax] / y0[rmax] /. NDSolve[{v'[r] + (d - 1) v[r]/r + a/ThetaGen[p, d]
        Abs[u[r]]^(p-2) u[r] - (1 - ThetaGen[p, d])/ThetaGen[p, d] u[r] == 0, u'[r] == v[r],
      y0'[r] == r^(d-1) u[r], y2'[r] == r^(d+1) u[r], v[ε] == -(a + ThetaGen[p, d] - 1)/ThetaGen[p, d] ε/d,
      u[ε] == 1 - (a + ThetaGen[p, d] - 1)/ThetaGen[p, d] ε^2/(2 d), y0[ε] == ε^d/d - (a + ThetaGen[p, d] - 1)/ThetaGen[p, d] ε^(d+1)/(d (d + 1)),
      y2[ε] == ε^(d+2)/(d + 2) - (a + ThetaGen[p, d] - 1)/ThetaGen[p, d] ε^(d+3)/(d (d + 3))},
     {u, v, y0, y2}, {r, ε, rmax}]] [[1]]

Iter[a_, h_, p_, d_, rmax_, ε_, b_, η_, j_, Nmax_] :=
 Module[{M = H[a + h, p, d, rmax, ε]}, If[Or[Abs[b - M] < η, j > Nmax],
   {j, N[a], M, M - b, N[h], IGN[p, d, a, rmax, ε], p, K[ThetaGen[p, d], p]},
   If[M < b, Iter[a + h, h, p, d, rmax, ε, M, η, j + 1, Nmax],
    Iter[a + h, -h / 2, p, d, rmax, ε, M, η, j + 1, Nmax]]]]

Init[a_, h_, p_, d_, rmax_, ε_, η_, Nmax_] :=
 Iter[a, h, p, d, rmax, ε, H[a, p, d, rmax, ε], η, 1, Nmax]

ResultGN[a_, h_, p_, d_, rmax_, ε_, η_, amin_, amax_, Nmax_] :=
 Module[{M = Init[a, h, p, d, rmax, ε, η, Nmax]}, {M[[6]], t[M[[6]], p, d, rmax, ε]}]

Delta2 = Plot[ResultGN[1, 1, p, 5, 7.5176, 10^-12, 10^-15, 1, 20, 100] [[1]] S[5]^((p-2)/p) -
    1 / Cstar[p, 5, LambdaFS[p, ThetaGen[p, 5], 5], ThetaGen[p, 5]],
   {p, 2.8, 3.3}, PlotStyle → {Dashing[Tiny], Thickness[0.005], Black}];
```

```
Show[Delta1, Delta2, PlotRange → All]
```