

Line Search Algorithm for Optimization

- Mathe 3 (CES)
- WS21
- Lambert Theisen (theisen@acom.rwth-aachen.de)

• `using PlutoUI` , `Calculus` , `Gadfly` , `LinearAlgebra`

Define Objective

$$f(x) = x^2$$

`f = #1 (generic function with 1 method)`

• `f = (x -> x[1]^2)`

Line Search

1. Given $x^{(0)}$
2. For $k = 0, 1, 2, \dots$ do
 1. Update: $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$
3. End

`line_search (generic function with 1 method)`

```
• function line_search(f, x0, α, d, kmax)
•     x = x0
•     hist = []
•     push!(hist, x)
•     for k=1:kmax
•         x = x + α(x) * d(x)
•         push!(hist, x)
•     end
•     return x, hist
• end
```

Check Line Search

- Observe that different step sizes change the result!

```
(0.0, [1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0])
```

```
• line_search(f, 1, (x->1.0), (x->-sign(x)), 10)
```

Gradient Descent

- Is line search with $d^{(k)} = -\nabla f(x^{(k)})$

hessian (generic function with 9 methods)

```
• begin
•   # some notation
•   ∇ = derivative
•   ∇² = hessian
• end
```

gradient_descent (generic function with 1 method)

```
• function gradient_descent(f, x0, α, kmax)
•   return line_search(f, x0, α, (x->-∇(f, x)), kmax)
• end
```

Check Gradient Descent

```
(
  1: 2.03704e-10
  2: [1, 0.8, 0.64, 0.512, 0.4096, 0.32768, 0.262144, 0.209715, 0.167772, more, 2.0
)
```

```
• gradient_descent(f, 1, (x->0.1), 100)
```

```
(2.65614e-5, [1, -0.9, 0.81, -0.729, 0.6561, -0.59049, 0.531441, -0.478297, 0.430467,
```

```
• gradient_descent(f, 1, (x->0.95), 100) # slower, oscillating but converging
```

Newton's Method for Optimization

- Is line search with $d^{(k)} = -[\nabla^2 f(x^{(k)})]^{-1} \nabla f(x^{(k)})$

```
newton (generic function with 1 method)
```

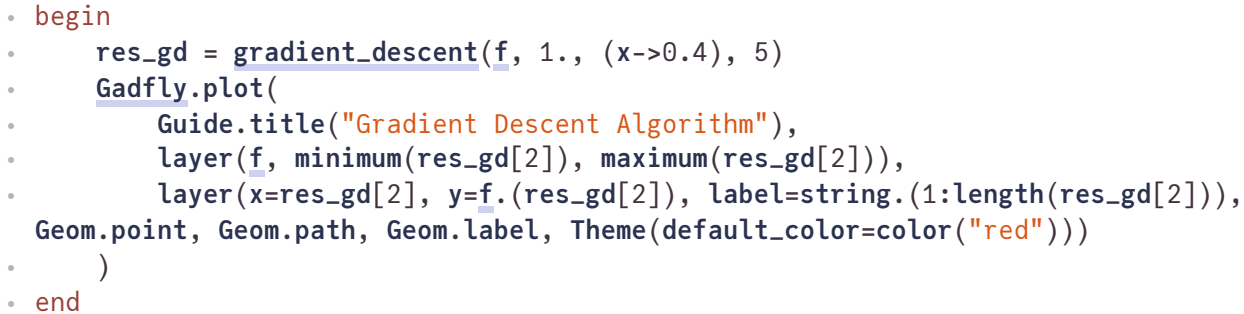
- `function newton(f, x0, α, kmax)`
- `return line_search(f, x0, α, (x->-inv(∇2(f, x))*∇(f, x)), kmax)`
- `end`

Check Newton's Method

```
(1.05879e-22, [1.0, -7.28306e-7, 1.05879e-22, 1.05879e-22, 1.05879e-22, 1.05879e-22, 1.05879e-22, 1.05879e-22, 1.05879e-22, 1.05879e-22])
```

- `newton(f, 1., (x->1.0), 100) # works well 😎`

```
(1.26764e30, [1.0, -2.0, 4.00001, -7.99999, 16.0, -31.9999, 63.9998, -128.0, 255.999, -512.0, 1023.99, -2048.0, 4096.0, -8192.0, 16384.0, -32768.0, 65536.0, -131072.0, 262144.0, -524288.0, 1048576.0, -2097152.0, 4194304.0, -8388608.0, 16777216.0, -33554432.0, 67108864.0, -134217728.0, 268435456.0, -536870912.0, 1073741824.0, -2147483648.0, 4294967296.0, -8589934592.0, 17179869184.0, -34359738368.0, 68719476736.0, -137438953472.0, 274877906944.0, -549755813888.0, 1099511627776.0, -2199023255552.0, 4398046511104.0, -8796093022208.0, 17592186044416.0, -35184372088832.0, 70368744177664.0, -140737488355328.0, 281474976710656.0, -562949953421312.0, 1125899906842624.0, -2251799813685248.0, 4503599627370496.0, -9007199254740992.0, 18014398509481984.0, -36028797018963968.0, 72057594037927936.0, -144115188075855872.0, 288230376151711744.0, -576460752303423488.0, 1152921504606846976.0, -2305843009213693952.0, 4611686018427387904.0, -9223372036854775808.0, 18446744073709551616.0, -36893488147419103232.0, 73786976294838206464.0, -147573952589676412928.0, 295147905179352825856.0, -590295810358705651712.0, 1180591620717411303424.0, -2361183241434822606848.0, 4722366482869645213696.0, -9444732965739290427392.0, 18889465931478580854784.0, -37778931862957161709568.0, 75557863725914323419136.0, -151115727451828646838272.0, 302231454903657293676544.0, -604462909807314587353088.0, 1208925819614629174706176.0, -2417851639229258349412352.0, 4835703278458516698824704.0, -9671406556917033397649408.0, 19342813113834066795298816.0, -38685626227668133590597632.0, 77371252455336267181195264.0, -154742504910672534362390528.0, 309485009821345068724781056.0, -618970019642690137449562112.0, 1237940039285380274899124224.0, -2475880078570760549798248448.0, 4951760157141521099596496896.0, -9903520314283042199192993792.0, 19807040628566084398385987584.0, -39614081257132168796771975168.0, 79228162514264337593543950336.0, -158456325028528675187087900672.0, 316912650057057350374175801344.0, -633825300114114700748351602688.0, 1267650600228229401496703205376.0, -2535301200456458802993406410752.0, 5070602400912917605986812821504.0, -10141204801825835211973625643008.0, 20282409603651670423947251286016.0, -40564819207303340847894502572032.0, 81129638414606681695789005144064.0, -162259276829213363391578010288128.0, 324518553658426726783156020576256.0, -649037107316853453566312041152512.0, 1298074214633706907132624082305024.0, -2596148429267413814265248164610048.0, 5192296858534827628530496329220096.0, -10384593717069655257060992658440192.0, 20769187434139310514121985316880384.0, -41538374868278621028243970633760768.0, 83076749736557242056487941267521536.0, -166153499473114484112975882535043072.0, 332306998946228968225951765070086144.0, -664613997892457936451903530140172288.0, 1329227995784915872903807060280344576.0, -2658455991569831745807614120560689152.0, 5316911983139663491615228241121378304.0, -10633823966279326983230456482242756608.0, 21267647932558653966460912964485513216.0, -42535295865117307932921825928971026432.0, 85070591730234615865843651857942052864.0, -170141183460469231731687303715884105728.0, 340282366920938463463374607431768211456.0, -680564733841876926926749214863536422912.0, 1361129467683753853853498429727072845824.0, -2722258935367507707706996859454145691648.0, 5444517870735015415413993718908291383296.0, -10889035741470030830827987437816582766592.0, 21778071482940061661655974875633165533184.0, -43556142965880123323311949751266331066368.0, 87112285931760246646623899502532662132736.0, -174224571863520493293247799005065324265472.0, 348449143727040986586495598010130648530944.0, -696898287454081973172991196020261297061888.0, 1393796574908163946345982392040522594123776.0, -2787593149816327892691964784081045188247552.0, 5575186299632655785383929568162090376495104.0, -11150372599265311570767859136324180752990208.0, 22300745198530623141535718272648361505980416.0, -44601490397061246283071436545296723011960832.0, 89202980794122492566142873090593446023921664.0, -178405961588244985132285746181186892047843328.0, 356811923176489970264571492362373784095686656.0, -713623846352979940529142984724747568191373312.0, 1427247692705959881058285969449495136382746624.0, -2854495385411919762116571938898990272765493248.0, 5708990770823839524233143877797980545530986496.0, -11417981541647679048466287755595961091061972992.0, 22835963083295358096932575511191922182123945984.0, -45671926166590716193865151022383844364247891968.0, 91343852333181432387730302044767688728495783936.0, -182687704666362864775460604089535377456991567872.0, 365375409332725729550921208179070754913983135744.0, -730750818665451459101842416358141509827966271488.0, 1461501637330902918203684832716283019655932542976.0, -2923003274661805836407369665432566039311865085952.0, 5846006549323611672814739330865132078623730171904.0, -11692013098647223345629478661730264157247460343808.0, 23384026197294446691258957323460528314494920687616.0, -46768052394588893382517914646921056628989841375232.0, 93536104789177786765035829293842113257979682750464.0, -187072209578355573530071658587684226515959365500928.0, 374144419156711147060143317175368453031918731001856.0, -748288838313422294120286634350736906063837462003712.0, 1496577676626844588240573268701473812127674924007424.0, -2993155353253689176481146537402947624255349848014848.0, 5986310706507378352962293074805895248510699696029696.0, -11972621413014756705924586149611790497021399392059392.0, 23945242826029513411849172299223580994042798784118784.0, -47890485652059026823698344598447161988085597568237568.0, 95780971304118053647396689196894323976171195136475136.0, -191561942608236107294793378393788647952342390272950272.0, 383123885216472214589586756787577295904684780545900544.0, -766247770432944429179173513575154591809369561091801088.0, 1532495540865888858358347027150309183618739122183602176.0, -3064991081731777716716694054300618367237478244367204352.0, 6129982163463555433433388108601236734474956488734408704.0, -12259964326927110866866776217202473468949912977468817408.0, 24519928653854221733733552434404946937899825954937634816.0, -49039857307708443467467104868809893875799651909875269632.0, 98079714615416886934934209737619787751599303819750539264.0, -196159429230833773869868419475239575503198607639501078528.0, 392318858461667547739736838950479151006397215279002157056.0, -784637716923335095479473677900958302012794430558004314112.0, 1569275433846670190958947355801916604025588861116008628224.0, -3138550867693340381917894711603833208051177722232017256448.0, 6277101735386680763835789423207666416102355444464034512896.0, -12554203470773361527671578846415332832204710888928069025792.0, 25108406941546723055343157692830665664409421777856138051584.0, -50216813883093446110686315385661331328818843555712276103168.0, 100433627766186892221372630771322662657637687111424552206336.0, -200867255532373784442745261542645325315275374222849104412672.0, 401734511064747568885490523085290650630550748445698208825344.0, -803469022129495137770981046170581301261101496891396417650688.0, 1606938044258990275541962092341162602522202993782792835301376.0, -3213876088517980551083924184682325205044405987565585670602752.0, 6427752177035961102167848369364650410088811975131171341205504.0, -12855504354071922204335696738729300820177623950262342682411008.0, 25711008708143844408671393477458601640355247900524685364822016.0, -51422017416287688817342786954917203280710495801049370729644032.0, 102844034832575377634685573909834406561420991602098741459288064.0, -205688069665150755269371147819668813122841983204197482918576128.0, 411376139330301510538742295639337626245683966408394965837152256.0, -822752278660603021077484591278675252491367932816789931674304512.0, 1645504557321206042154969182557350504982735865633579863348609024.0, -3291009114642412084309938365114701009965471731267159726697218048.0, 6582018229284824168619876730229402019930943462534319453394436096.0, -13164036458569648337239753460458804039861886925068638906788872192.0, 26328072917139296674479506920917608079723773850137277813577744384.0, -52656145834278593348959013841835216159447547700274555627155488768.0, 105312291668557186697918027683670432318895095400549111254310977536.0, -210624583337114373395836055367340864637790190801098222508621955072.0, 421249166674228746791672110734681729275580381602196445017243910144.0, -842498333348457493583344221469363458551160763204392890034487820288.0, 1684996666696914987166688442938726917102321526408785780068975640576.0, -3369993333393829974333376885877453834204643052817571560137951281152.0, 6739986666787659948666753771754907668409286105635143120275902562304.0, -13479973333575319897333507543509815336818572211270286240551805124608.0, 26959946667150639794667015087019630673637144422540572481103610249216.0, -53919893334301279589334030174039261347274288845081144962207220498432.0, 107839786668602559178668060348078522694548577690162289924414440996864.0, -215679573337205118357336120696157045389097155380324579848828881993728.0, 431359146674410236714672241392314090778194310760649159697657763987456.0, -862718293348820473429344482784628181556388621521298319395315527974912.0, 1725436586697640946858688965569256363112777243042596638790631055949824.0, -3450873173395281893717377931138512726225554486085193277581262111899648.0, 6901746346790563787434755862277025452451108972170386555162524223799296.0, -13803492693581127574869511724554050904902217944340773110325048447598592.0, 27606985387162255149739023449108101809804435888681546220650096895197184.0, -55213970774324510299478046898216203619608871777363092441300193790394368.0, 110427941548649020598956093796432407239217743554726184882600387580788736.0, -220855883097298041197912187592864814478435487109452369765200775161577472.0, 441711766194596082395824375185729628956870974218904739530401550323154944.0, -883423532389192164791648750371459257913741948437809479060803100646309888.0, 1766847064778384329583297500742918515827483896875618958121606201292619776.0, -3533694129556768659166595001485837031654967793751237916243212402585239552.0, 7067388259113537318333190002971674063309935587502475832486424805170479104.0, -14134776518227074636666380005943348126619871175004951664972849610340958208.0, 28269553036454149273332760011886696253239742350009903329945699220681916416.0, -56539106072908298546665520023773392506479484700019806659891398441363832832.0, 113078212145816597093331040047546785012958969400039613319782796882727665664.0, -226156424291633194186662080095093570025917938800079226639565593765455331328.0, 452312848583266388373324160190187140051835877600158453279131187530910662656.0, -904625697166532776746648320380374280103671755200316906558262375061821325312.0, 1809251394333065553493296640760748560207343510400633813116524750123642650624.0, -3618502788666131106986593281521497120414687020801267626233049500247285301248.0, 7237005577332262213973186563042994240829374041602535252466099000494570602496.0, -14474011154664524427946373126085988481658748083205070504932198000989141204992.0, 28948022309329048855892746252171976963317496166410141009864396001978282409984.0, -57896044618658097711785492504343953926634992332820282019728792003956564819968.0, 115792089237316195423570985008687907853269984665640564039457584007913129639936.0, -231584178474632390847141970017375815706539969331281128078915168015826259279872.0, 463168356949264781694283940034751631413079938662562256157830336031652518559744.0, -926336713898529563388567880069503262826159877325124512315660672063305037119488.0, 1852673427797059126777135760139006525652319754650249024631321344126610074238976.0, -3705346855594118253554271520278013051304639509300498049262642688253220148477952.0, 7410693711188236507108543040556026102609279018600996098525285376506440296955904.0, -14821387422376473014217086081112052205218558037201992197050570753012880593911808.0, 29642774844752946028434172162224104410437116074403984394101141506025761187823616.0, -59285549689505892056868344324448208820874232148807968788202283012051522375647232.0, 118571099379
```



Define Objective

```
g = #23 (generic function with 1 method)
• g = (x->x[1]^2+x[2]^2)
```

- both work

```
res_n_2d =
(
    1:    [2.33348e-22, 3.04878e-22]
    2:    [[1.0, 1.0], [0.6, 0.6], [0.36, 0.36], [0.216, 0.216], [0.1296, 0.1296], [0.0777
)]
• res_n_2d = newton(g, [1.,1.], (x->0.4), 100)
```

```
[2.33348e-22, 3.04878e-22]
```

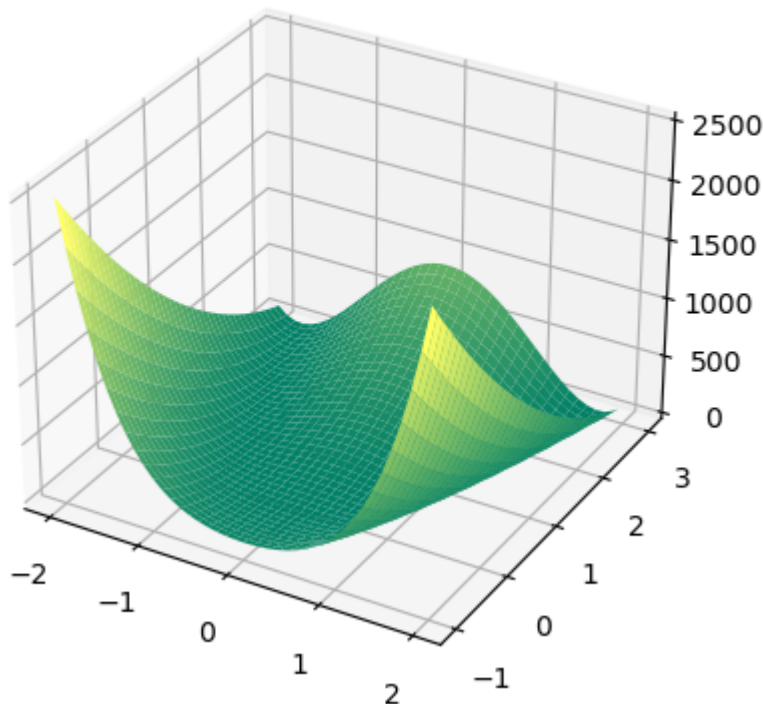
```
• res_n_2d[2][end]
```

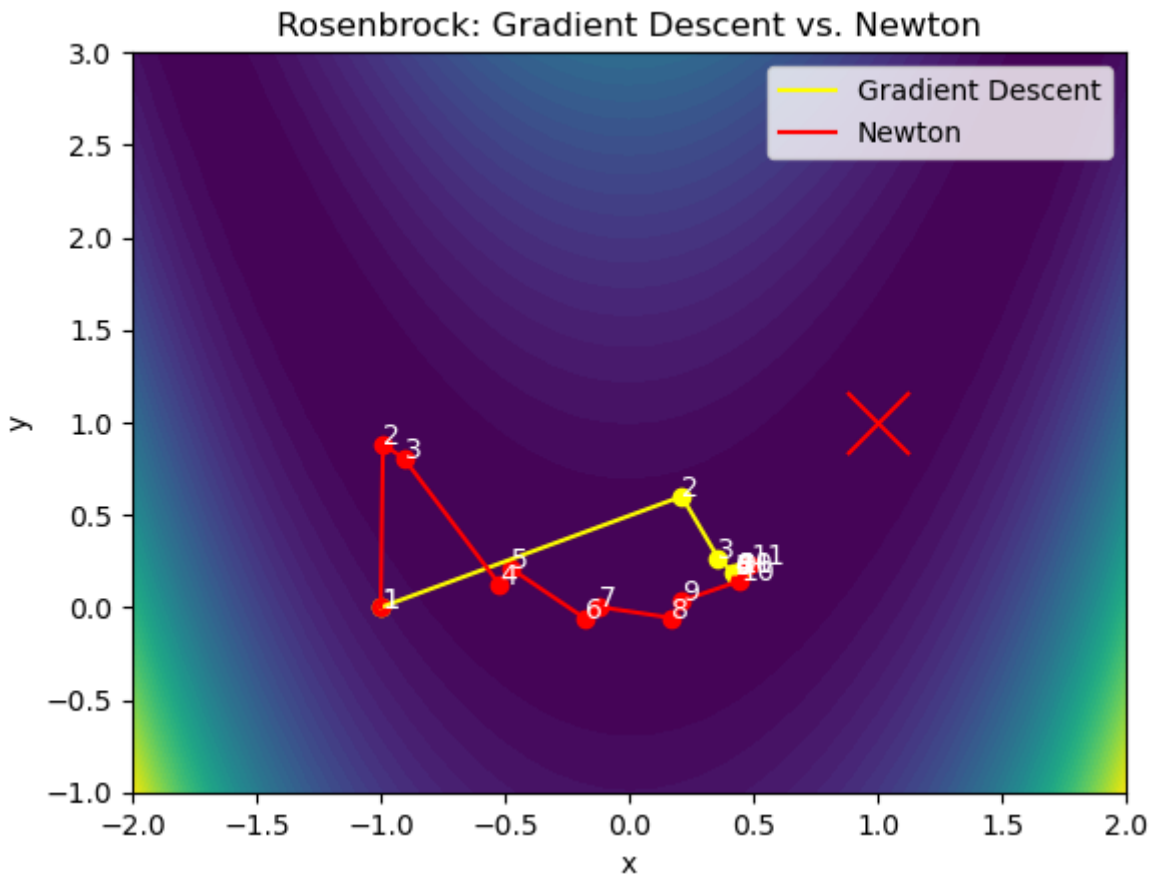
```
true
```

```
• norm(res_n_2d[2][end] - [0,0]) < eps(Float64) # is converged to machine-precision?
```

Test Gradient Descent vs Newton for 2D Rosenbrock

Rosenbrock Function





```

• begin
•   # Rosenbrock function with  $x^* = [a, a^2]$ ,  $f(x^*)=0$ 
•   a = 1
•   b = 100
•   h = (x -> (a-x[1])^2 + b*(x[2]-x[1]^2)^2)
•
•   x0 = [-1., 0.]
•
•   # Gradient Descent
•   res_gd_2d_rb = gradient_descent(h, x0, (x->0.003), 10)
•   res_gd_2d_rb_x = [res_gd_2d_rb[2][i][1] for i=1:length(res_gd_2d_rb[2])]
•   res_gd_2d_rb_y = [res_gd_2d_rb[2][i][2] for i=1:length(res_gd_2d_rb[2])]
•
•   # Newton
•   res_n_2d_rb = newton(h, x0, (x->0.9), 10)
•   res_n_2d_rb_x = [res_n_2d_rb[2][i][1] for i=1:length(res_n_2d_rb[2])]
•   res_n_2d_rb_y = [res_n_2d_rb[2][i][2] for i=1:length(res_n_2d_rb[2])]
•
•   clf()
•   Δ = 0.1
•   X=collect(-2:Δ:2)
•   Y=collect(-1:Δ:3)
•   F=[h([X[j],Y[i]]) for i=1:length(X), j=1:length(Y)]
•   contourf(X,Y,F, levels=50)
•   PyPlot.title("Rosenbrock: Gradient Descent vs. Newton")
•
•   # res_gd_2d_rb
•   PyPlot.plot(res_gd_2d_rb_x, res_gd_2d_rb_y, color="yellow")
•   scatter(res_gd_2d_rb_x, res_gd_2d_rb_y, color="yellow")
•   for i=1:length(res_gd_2d_rb_x)
•       annotate(string(i), [res_gd_2d_rb_x[i], res_gd_2d_rb_y[i]], color="w",
•           zorder=2)

```

```

•     end
•
•     # res_n_2d_rb
•     PyPlot.plot(res_n_2d_rb_x, res_n_2d_rb_y, color="red")
•     scatter(res_n_2d_rb_x, res_n_2d_rb_y, color="red")
•     for i=1:length(res_n_2d_rb_x)
•         annotate(string(i), [res_n_2d_rb_x[i], res_n_2d_rb_y[i]], color="w",
• zorder=2)
•     end
•
•     legend(["Gradient Descent", "Newton"])
•
•     xlabel("x")
•     ylabel("y")
•
•     # Mark minimum
•     scatter(a, a^2, color="r", s=500, zorder=3, marker="x")
•
•     gcf()
end

```

Broyden's Method

Homework: Adapt GD and Newton to use the generic framework

line_search2 (generic function with 1 method)

```

• function line_search2(f, x0, α, B0, Bk, kmax, tol)
•     x = x0
•     B = B0
•     k = 0
•     Δx = Inf
•     hist = []
•     push!(hist, x)
•     while (k <= kmax) && (norm(Δx) > tol)
•         # invB = length(x)==1 ? 1/B
•         d = -inv(B) * ∇(f, x)
•         Δx = α(x) * d
•         x = x + Δx
•         B = Bk(x, d, f, B)
•         push!(hist, x)
•         k += 1
•     end
•     return x, hist
• end

```

broyden (generic function with 1 method)

```

• function broyden(f, x0, α, kmax, tol)
•     return line_search2(
•         f, x0, α,
•         I(length(x0)),
•         (x,d,f,B)->(B + (∇(f, x) * d') / (norm(d,2)^2)), kmax, tol
•     )
• end

```

- `broyden(g, [1.,1.], (x->0.4), 100, 1E-10)` *# works quite fast*

- `newton(g, [1.,1.], (x->0.4), 100)` # slower than Broyden 🐢

- `broyden(f, [1], (x->0.1), 100, 1E-10)` *# works*