AD-Kompaktkurs in Decin

Aufgaben zu den Modellprobleme (ADMAT)

12. M"arz:

0. Installation of ADMAT

- Unpack ADMAT.tar.gz into any folder (say /home/user/john/ADMAT). `gunzip -c ADMAT.tar.gz | tar -xv`.
- Run Matlab and change to above directory and execute startup.
- ADMAT is now installed and you are ready to run the examples. (If you got any warnings during the startup then there might be problems. Please ask the instructor)

Course Files

All the files are available in ADMAT/COURSE directory.

1. Trigger-Schaltkreis

Forward differentiation of the trigger function.

The function is available in the file trigger_fun.m

Exercise: F=trigger_fun(x,Extra) computes the function value where Extra is constructed as from parameters as a matlab structure.

Extra.prm=prm; where prm are the parameters of the problem; prm = [5.7 1.2];

Try the following values for the independents: x = [5.1 ; 5.7 ; 5.7 ; 5.5];

Compute the derivative in the forward mode using the forwprod routine with the following arguments. (See help forwprod for help)

[f,JV]=forwprod('trigger_fun',x,V,[],Extra);

This commands are available in file trigger_driver.m for you. But, please try to run the ADMAT commands on your own before looking at the usage in the driver file.

2. Stabilit"atsanalyse einer Kugel mit exzentrischem Schwerpunkt

Forward differentiation of the Kreisel function.

The function is available in the file kreisel_fun.m.

F=kreisel_fun(x,Extra) computes the function value where Extra.th = th is the parameter.

Compute the derivative in the forward mode using the forwprod routine with the following arguments.

This is available in file kreisel_forward.m for your use. Try comparing the value of the forward derivatives to the ones obtained by finite differencing.

3. Zahnradkräfte

Forward differentiation of the zahnkraefte function.

The function is available in the file zahn_fun.m.

F=zahn_fun(x,Extra) computes the function value where Extra is constructed as Extra.cd=cdval;Extra.parmax=parmax. cdval and parmax are parameters (default parmax=2 and cdval=[0.1 3.0 1.0/3.0])

Lets try the following values for the independents: x = [0 1];

Compute the derivative in the forward mode, using the forwprod routine with the following arguments.

This code is available in file zahn_forward.m for you to look at.

5. Simulation eines Brusselators

Forward differentiation of the Brusselators function.

This function implements the model of a brusselator.

The function is available in the file brus_fun.m.

F=brus_fun(x,Extra) computes the function value where Extra is constructed as Extra.prm=prm;Extra.h=h;Extra.dim=dim;

(Use parmax=6 and dim = 84 and h = 2.0/(dim+2.0) as the parameters of the problem.)

Then compute the derivative in the forward mode, we can use the forwprod routine with the following arguments.

Please try the independents value in file brus_driver.m . You can also find parameter values and the usage of forwprod command there.

13. M"arz:

4. Explosionsgleichung

Reverse differentiation of the Explosion function.

This function implements the discretized differentiatial equation describing an explosion.

The function is available in the file expl_fun.m

F=expl_fun(x,Extra) computes the function value where Extra is constructed as Extra.prm=prm;Extra.h=h;Extra.dim=dim; (use im = 7; parmax = 2; prm = [1.3 0.245828];h = 2.0/(dim+1.0);)

Try the following value of x x=[1.7200 ;3.4500 ; 4.1600 ; 4.8700 ; 4.1600 ;3.4500 ; 1.7200];

The compute the derivative in the reverse mode, using the revprod routine with the following arguments. (See help revprod for help) Compute the full Jacobian, by choosing an appropriate W in the command:
[f,WJ]=revprod('expl_fun',x,W,Extra);

Please try to run the ADMAT commands on your own before looking at the usage in the driver file expl_driver.m . Also assess the space requirements by running the matlab command spacerequired

2. Stabilit"atsanalyse einer Kugel mit exzentrischem Schwerpunkt

Reverse differentiation of the Kreisel function.

The function is available in the file kreisel_fun.m.
F=kreisel_fun(x,Extra) computes the function value where Extra.th = th is the parameter.

Compute the derivative in the reverse mode using the revprod routine.
This is available in file kreisel_reverse.m for your use. Try comparing the value of the derivatives to the ones obtained by finite differencing. Also assess the space requirements by running the matlab command spacerequired

6. Tragfähigkeitsberechnung von Radialgleitlagern

Reverse differentiation of the Ringlager function.

The function is available in the file ringlager_fun.m.
F=ringlager_fun(x,Extra) computes teh function value

Extra collects all the parameters in a structure Extra.ep=ep; Extra.b=b; Extra.So=So where you can use ep = 0.5 b = 0.4 So = 0.5.
try the following values for the independents (constructed from parameters):

$$x(1) = 1.1642 - 1.9456 * b + 7.1161 * b * b - 10.1073 * b^3 + 5.0141 * b^4;$$
$$x(2) = -1.000026 - 0.23634 * b - 0.4215 * b * b - 0.038817 * b^3 - 0.090551 * b^4;$$

The compute the derivative in the reverse mode, we can use the revprod routine with the following arguments.

[e,WJ] = revprod('ringlager_fun',x,W,Extra);

This code is available in file ringlager_driver.m for your use. Try different values of W and compare results with forward mode and finite difference. Also assess the space requirements by running the matlab command spacerequired.

7. Gasaustausch in zwei Hemisphären

Reverse differentiation of the gas model

This function does the integration of an discretized ODE describing the exchange of two gases in a two-box-model.

The function is available in the file bmod_fun.m

F=bmod_fun(x,Extra)

Extra collects all the parameters in a structure Extra.ny=ny; Extra.ntpy=ntpy; Extra.m=m; Extra.n =n; where ny=10; ntpy=50;

load independent variables from the file by issuing the command: load src_CH3CCl3.d;

The compute the derivative in the reverse mode, we can use the revprod routine with the following arguments.

This is available in file bmod_driver.m. Try to see the space requirements using function spacerequired.

8. Komplexer Quelltexte: A financial example

Reverse differentiation of the financial option pricing function.

This compute sensitivity of option prices to underlying stock price changes. The option prices are computed by the famous Black-Scholes formula which is differentiated by ADMAT.The option pricing function is available in the file myblsprice.m and the driver routine in blsprice_driver.m.

[CALL,PUT] = myblsprice(SO,X,R,T,SIG,Q) returns the value of call and put options using the Black-Scholes pricing formula. SO is the current

asset price, X is the exercise price, R is the risk-free interest rate, T is the time to maturity of the option in years, SIG is the standard deviation of the annualized continuously compounded rate of return of the asset (also known as volatility), and Q is the dividend rate of the asset.

For example, the current price an asset is 100, the exercise price of the option is 95, the risk-free interest rate is 10to maturity of the option is .25 years, and the standard deviation of of the asset is 50

call = myblsprice(100,95,.1,.25,.5,0)

Please run the file blsprice_driver to see the sensitivity computation.

Exercise:

- Convert the given myblsprice function into ADMAT format function assuming only the stock price is the underlying.
- Then Compute the delta or the gradient of call option values at underlying price 100, strike price=[80;90;100;110;120] and maturities =[1;1.5;2;2.5;3] years (use Volatility =0.3, Interest rate = 0.05 and dividend =0.02)
- Use both reverse and forward mode (i.e. revprod and forwprod routines)
- Compute the Gamma of the call option values by running HtimesV command from ADMIT (see help HtimesV) for usage.

9. ADMIT-1 demo[1ex]

Please run the slideshow by issuing the command from matlab command window : admitdemo This will illustrate the sparse Jacobian computation via ADMIT using the bi-coloring scheme of Coleman and Verma.

Then try evalJ commands on any of the above 8 functions to compute sparse Jacobians.