DarkSUSY 6

Tutorial

Joakim Edsjö
edsjo@fysik.su.se

With Torsten Bringmann, Paolo Gondolo, Piero Ullio and Lars Bergström

Stockholm University

TOOLS Conference
170911

Oscar Klein centre

DarkSUSY 6 pre-release 1

- This is a pre-release. Compared to the final DS 6.0 version (expected later this fall):
  - It is not fully finalized regarding SLHA reading/writing.
  - It is not fully tidied up and commented (output statements, main programs e.g.)
  - It does not have a completely updated manual
  - It is not finalized regarding charged cosmic ray diffusion and the interface to different halo models.
  - We have not yet tested on all compilers. gfortran 5 and 6 should work.
- If you find problems/have questions, e-mail edsjo@fysik.su.se

Please don’t distribute this pre-release version further at this point!

DarkSUSY 6 pre-release 1

- Download: www.astroparticle.se/ds/
- Unpack it: tar zxvf darksusy-6.0-pre1.tar.gz
- Replace examples/dsmain_wimp.F with the version on the web page above

```
./configure
make
```

Please don't distribute this pre-release version further at this point!

Outline of hands-on

1. dstest program
2. dsmain_wimp program
   - MSSM
   - generic WIMP, etc
   - how to use makefiles with DS 6
3. Replaceable functions
4. creating a new particle physics module
5. Other example programs

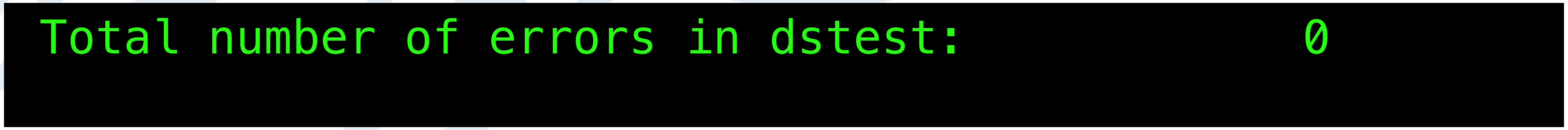
1. dstest program

- The dstest program is used to test your installation
- It calculated observables (masses, relic density, direct and indirect rates, …) and compares with pre-computed values

```
cd examples/test
./dstest
```

(already compiled with main make, takes about 60 seconds to run)

Output should end with



```
 Total number of errors in dstest:            0
```

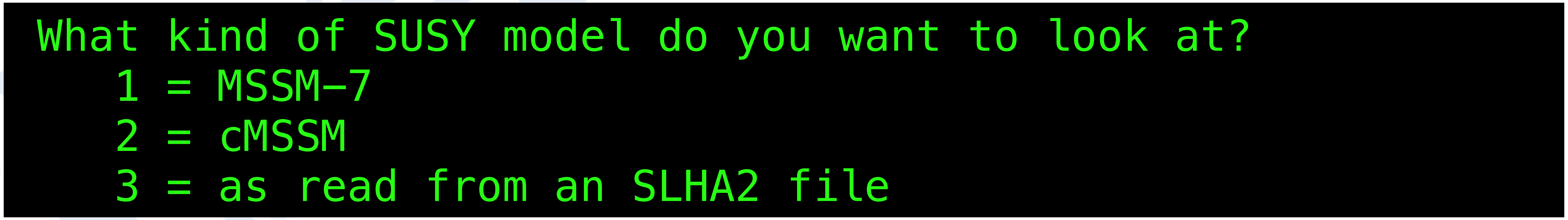
[Show code]

2. dsmain_wimp.F

- In examples/ we have the file dsmain_wimp.F which essentially does what dstest does, but in a more user-friendly way.
- run it with

```
./dsmain_wimp
```

- It will ask you which model you want to run:

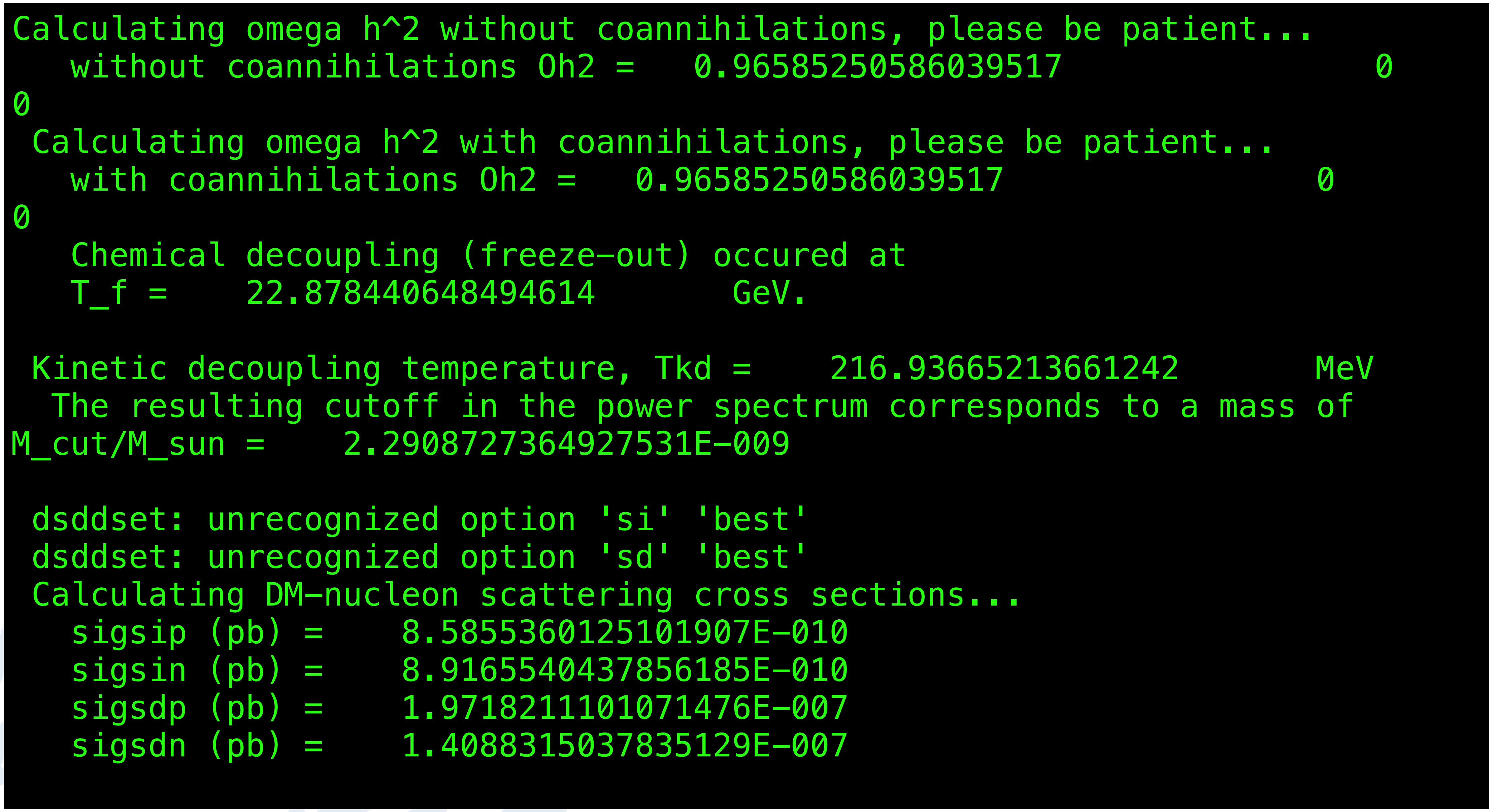


```
What kind of SUSY model do you want to look at?
    1 = MSSM-7
    2 = cMSSM
    3 = as read from an SLHA2 file
```

MSSM-7 example

- Pick 1: MSSM-7 and enter (e.g.)
  mu: 1000
  M2: 1000
  MA: 400
  tan(β): 10
  m0: 3000
  At/m0: 0
  Ab/m0: 0
- Then answer 0 to not write out an SLHA file (or something else if you want to)
- Observables are then calculated…

Output (cut)

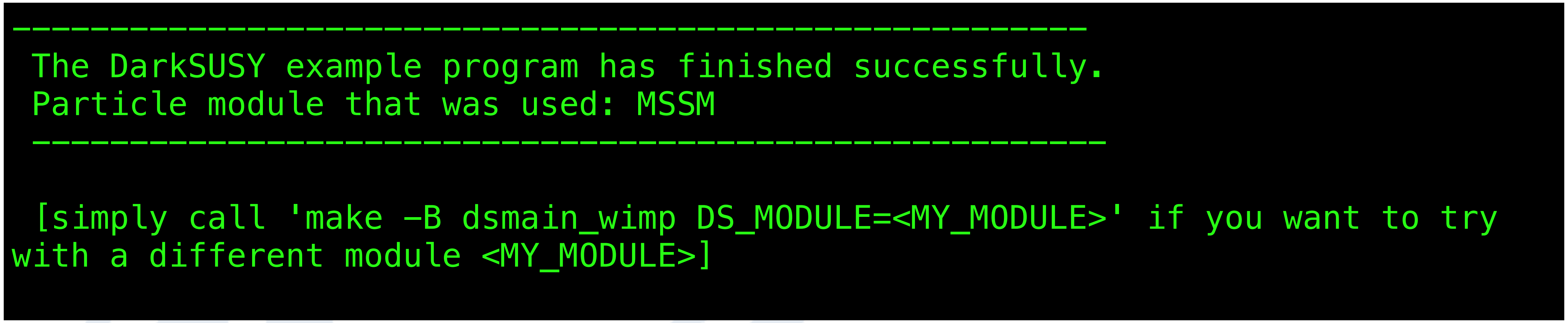


```
Calculating omega h^2 without coannihilations, please be patient...
   without coannihilations Oh2 =   0.96585250586039517                0
0
 Calculating omega h^2 with coannihilations, please be patient...
   with coannihilations Oh2 =   0.96585250586039517                0
0
   Chemical decoupling (freeze-out) occured at
   T_f =    22.878440648494614        GeV.

 Kinetic decoupling temperature, Tkd =    216.93665213661242       MeV
  The resulting cutoff in the power spectrum corresponds to a mass of
M_cut/M_sun =    2.2908727364927531E-009

 dsddset: unrecognized option 'si' 'best'
 dsddset: unrecognized option 'sd' 'best'
 Calculating DM-nucleon scattering cross sections...
   sigsip (pb) =    8.5855360125101907E-010
   sigsin (pb) =    8.9165540437856185E-010
   sigsdp (pb) =    1.9718211101071476E-007
   sigsdn (pb) =    1.4088315037835129E-007
```

etc

Which module?

- At the end of the dsmain_wimp run we got



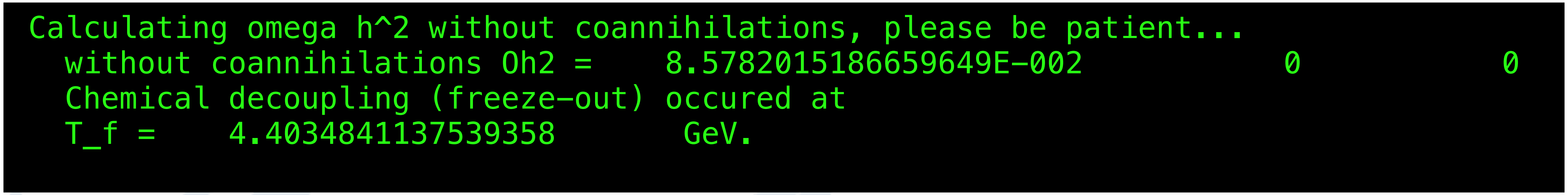
```
-------------------------------------------------------
 The DarkSUSY example program has finished successfully.
 Particle module that was used: MSSM
 -------------------------------------------------------

 [simply call 'make -B dsmain_wimp DS_MODULE=<MY_MODULE>' if you want to try
with a different module <MY_MODULE>]
```

- Try compiling again with

```
make -B dsmain_wimp DS_MODULE=generic_wimp
./dsmain_wimp
```

-
  ```
  Enter e.g.
  mass: 100
  self-conjugate: 0
  ann cross section: 3e-26
  PDG: 5
  scattering cross section: 1e-42
  ```

Output



```
Calculating omega h^2 without coannihilations, please be patient...
  without coannihilations Oh2 =    8.5782015186659649E-002           0           0
  Chemical decoupling (freeze-out) occured at
  T_f =    4.4034841137539358       GeV.
```

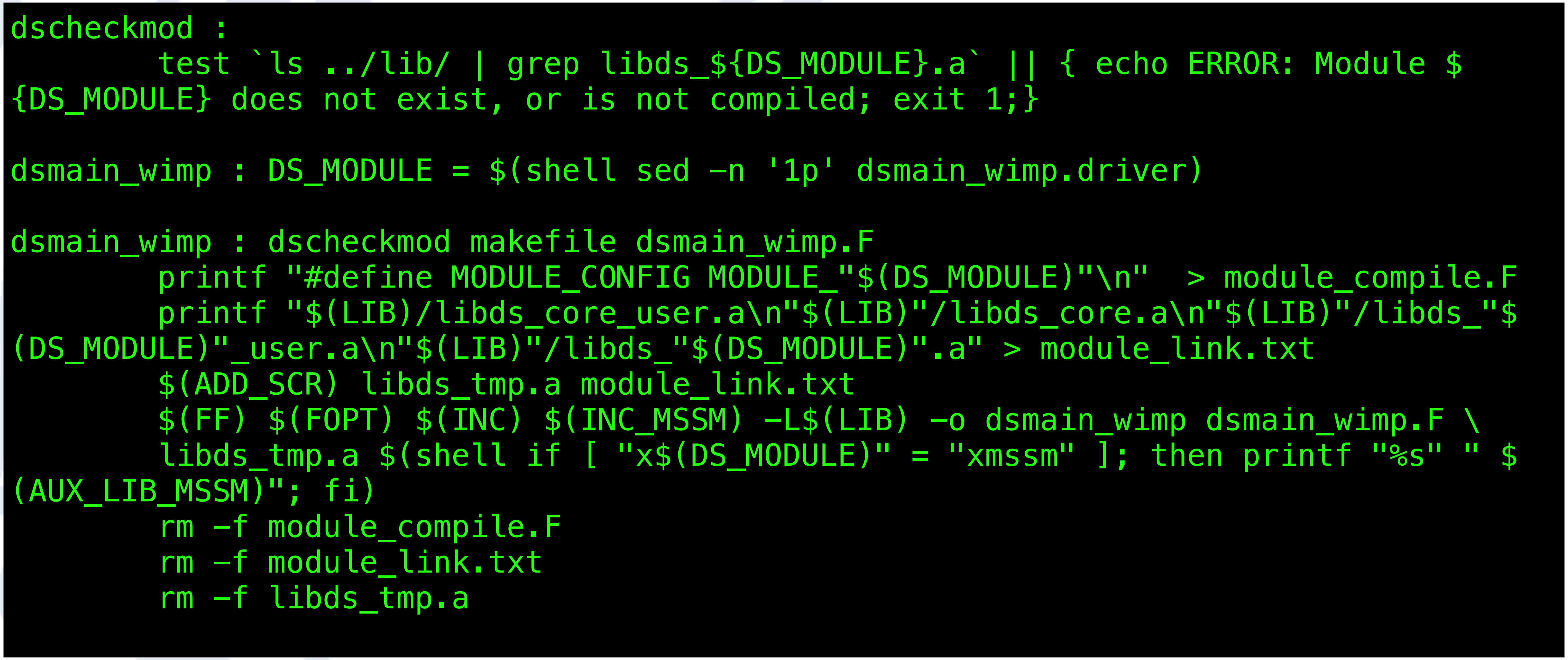
etc

Makefiles

- The way we choose which particle physics module to use is when we build our main program, e.g.

```
gfortran -o dsmain_wimp dsmain_wimp.F -lds_core.a -lds_mssm.a
```

- This can be made more flexible with makefiles,



```
dscheckmod :
        test `ls ../lib/ | grep libds_${DS_MODULE}.a` || { echo ERROR: Module $
{DS_MODULE} does not exist, or is not compiled; exit 1;}

dsmain_wimp : DS_MODULE = $(shell sed -n '1p' dsmain_wimp.driver)

dsmain_wimp : dscheckmod makefile dsmain_wimp.F
        printf "#define MODULE_CONFIG MODULE_"$(DS_MODULE)"\n"  > module_compile.F
        printf "$(LIB)/libds_core_user.a\n"$(LIB)"/libds_core.a\n"$(LIB)"/libds_"$
(DS_MODULE)"_user.a\n"$(LIB)"/libds_"$(DS_MODULE)".a" > module_link.txt
        $(ADD_SCR) libds_tmp.a module_link.txt
        $(FF) $(FOPT) $(INC) $(INC_MSSM) -L$(LIB) -o dsmain_wimp dsmain_wimp.F \
        libds_tmp.a $(shell if [ "x$(DS_MODULE)" = "xmssm" ]; then printf "%s" " $
(AUX_LIB_MSSM)"; fi)
        rm -f module_compile.F
        rm -f module_link.txt
        rm -f libds_tmp.a
```

dsmain_wimp.F

- dsmain_wimp.F is a good starting point for your own program. If you want to use it as a starting point,
  - make a copy out of it
  - modify examples/makefile.in to copy-paste the lines about dsmain_wimp.F and modify to your liking
  - run ./configure in the DS root
  - make and be happy!

Some details of dsmain_wimp.F

- In dsmain_wimp we have code blocks of this type

```
#if MODULE_CONFIG == MODULE_generic_wimp
     subroutine dspmenterparameters
     [more code for this module]
#endif
```

- This is how dsmain_wimp.F performs model-specific setup.
- We could as well have prepared one separate main program for each particle physics module if we preferred (the makefile is then a bit simpler as well, see e.g. examples/aux/makefile)

3. Replaceable functions

- If you want to modify an existing DarkSUSY function or subroutine, **DON'T**!
- Instead create your own version of the routine and link to that one instead.
- You can either just create your own version and link to it (before the DS library is linked to), or
- Use the script scr/make_replaceable.f to make a user_replaceable function for you, for which the makefiles are already set up to work

Replaceable function example

- As an example, we will look at the source term for DM annihilation in the galactic halo

$$\mathcal{S}_2(E_f) = \frac{1}{N_\chi m_\chi^2} \sum_i \sigma_i v \frac{dN_i}{dE_f} \,,$$

This code is in src_models/generic_wimp/dscrsource.f

- Let's add a boost factor from substructures

$$\mathcal{S}_2(E_f) = \frac{1}{N_\chi m_\chi^2} \sum_i \sigma_i v \frac{dN_i}{dE_f} \; {*B}$$

Replaceable function (cont)

- In the root directory, type
  ```
  scr/make_replaceable.pl src_models/
  generic_wimp/cr/dscrsource.f
  ```
- This will give you a new file src_models/generic_wimp/user_replaceables/dscrsource.f
- Modify it, configure and make again (in the root), then
  ```
  make -B dsmain_wimp DS_MODULE=generic_wimp
  ```
  in examples and run dsmain_wimp

4. Creating a new particle physics module

- To create a completely new particle physics module, either
  - write it from scratch, making sure to include the interface functions you need, or
  - start from an already existing particle physics module (will use this as an example)

Particle physics modules

- In src_models we currently have
  - mssm - Minimal Supersymmetric Standard Model
  - silveira_zee - Scalar singlet model
  - generic_wimp - a generic annihilating WIMP model
  - generic_decayingDM - a generic decaying dark matter model
  - empty - an empty model with just the basic set of interface functions for a ‘fresh’ start
- If you add one and want others to use it, please let us know and we can add it to the distribution (or point to your preferred download page)

Simple example, extend generic wimp

- Create a new module by typing (in the root directory)

  ```
  scr/make_module.pl generic_wimp extended_wimp
  ```

- Then type

  ```
  ./configure
  make
  ```

- You then have a new module extended_wimp in src_models
- It is right now identical to generic_wimp, but you can now modify it to your liking

You need to have autoconf installed for this to work

Helpful tools

- The extended_wimp is automatically included in the build system, but when/if you start adding files you need to tell the build system. To help you, we have a few scripts
  - scr/makemf.pl <directory> - adds all source files in the given <directory> to the relevant makefiles, or rather makefile.in's (without argument it adds source files in all directories in src/ and src_models)
  - scr/preconfig.pl - adds source files AND new directories to the build system and updates both the configure script and makefiles

You need to have autoconf installed for this to work

Main program

- You can e.g. use your new module with dsmain_wimp (or any other main program you choose)

- For dsmain_wimp, you need it to be aware of your new module by adding lines of this type:

  ```
  #if MODULE_CONFIG == MODULE_extended_wimp
  [add your code here]
  #endif
  ```

  This can be done by e.g. copy-pasting the corresponding generic_wimp lines and replace generic_wimp with extended_wimp

5. Other main programs

- In examples/aux we have a few example programs for other typical calculations, e.g.
  - the program to calculate the relic density in the Silveira Zee model
  - the program to calculate the relic density in the generic wimp model

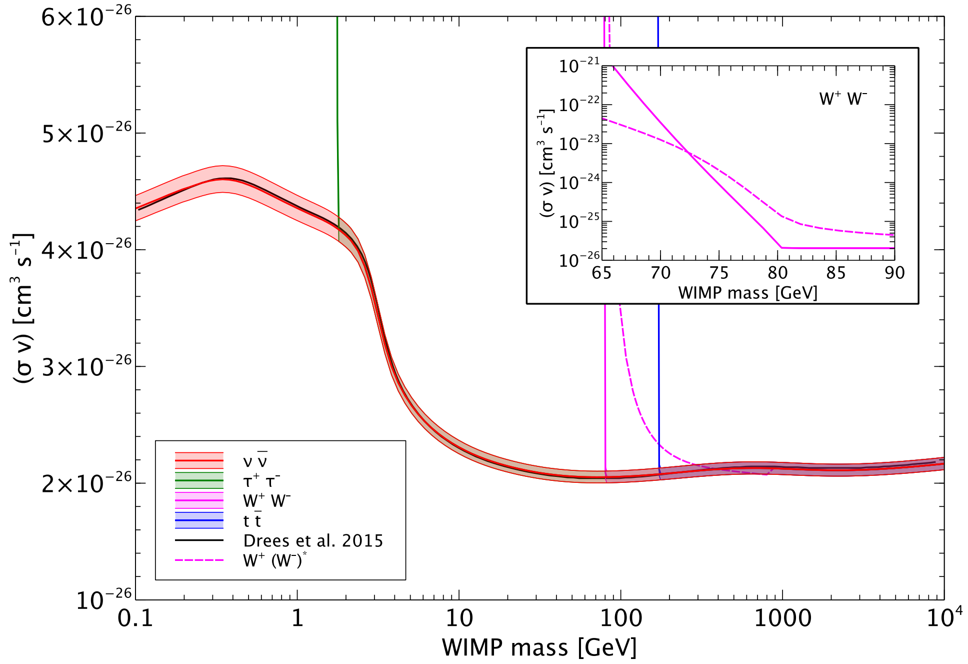
will look at this code

- we will add more examples and a description later

generic_wimp_oh2

- This is the example program that creates the figure on relic density

  ```
  cd examples/aux
  make generic_wimp_oh2
  ./generic_wimp_oh2
  ```



  Creates an output file generic_wimp_oh2-planck-sigmav-thr.dat that can e.g. be plotted

- It scans through the mass range, and for each mass makes a binary search in sigma v to find the Planck measurement ± 2 sigma

- The default setup takes about 11.5 min to run, change ‘f=1.1’ to ‘f=1.3’ in line 40 and ‘fth=1.02’ to ‘fth=1.1’ on line 41 to speed it up for the tutorial (takes 3m20s on my laptop)

Comment

- The default in generic_wimp is to use a sharp cut-off in W_{eff} when $m_X < m_{\mathrm{final}}$
- We can use an effective model with an off-shell final state particle, i.e. $XX \to W^{+} W^{-*}$
- An implementation of this is in examples/aux/user_replaceables/dsanwx.f
- Just compile replacing the regular dsanwx.f with this new one to test it:

```
make generic_wimp_oh2_threshold
```

Conclusions

- DarkSUSY 5 publically available
- DarkSUSY 6 is much more modular and include other improvements. Pre-release 1 available now. Expect full version later this fall
- When comparing different signals, it is crucial to perform these calculations in a consistent framework, with e.g. a tool like DarkSUSY

ευχαριστώ!

Stockholm University

Joakim Edsjö
edsjo@fysik.su.se

Oskar Klein centre