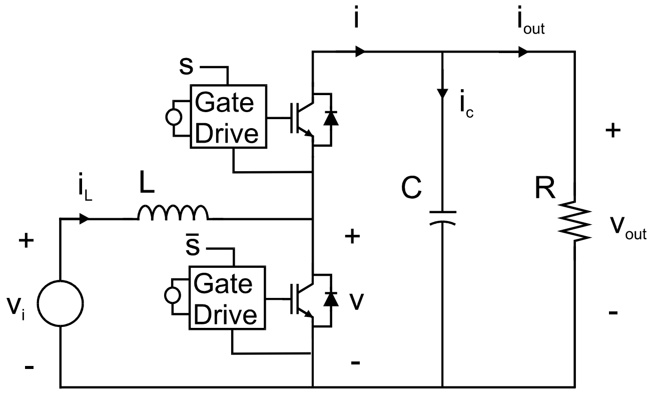
EE595S Notes - ACSL
Fall 2005

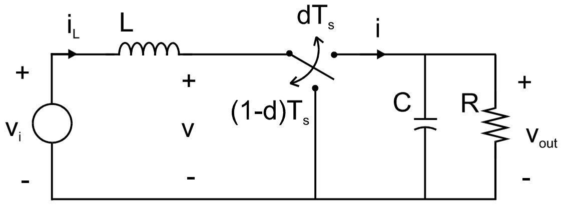
Largely Taken w/o Permission
From ESAC ACSL Course
Section 2:
Running ACSL Programs
Keith Corzine
Energy Systems Analysis Consortium
7/7/98 - 7/10/98

Example 1: DC/DC Converter

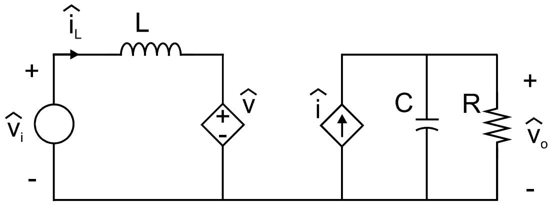


Average-Value Model

Switching Circuit



Replace Switching With Dependant Sources



$\hat{v} = d\hat{v}_o$ $\quad$ $\hat{i} = d\hat{i}_L$

^ denotes fast average

Average-Value Equations

State Equations

$$\begin{bmatrix} p\hat{i}_L \\ p\hat{v}_o \end{bmatrix} = \begin{bmatrix} 0 & -\frac{d}{L} \\ \frac{d}{C} & -\frac{1}{RC} \end{bmatrix} \begin{bmatrix} \hat{i}_L \\ \hat{v}_o \end{bmatrix} + \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix} v_i$$

ACSL

- ACSL is
  - A programming language for simulation of ODEs
- Consists of
  - A Builder
    - A translater
    - A compiler
  - A run time environment

ACSL Files

- Project File: *.PRJ (Created By Builder)
- Source Code: *.CSL (Created By You)
- Command File: *.CMD (Created By You)
- Macros: *.MAC (Created By You)
- Fortran and C Routines: (Created By You)
- Fortran Program *.F (Created By ACSL)
- Data File *.rrr (Created By Runtime)
- Log File *.log (Created By ACSL)
- Error File *.err (Created By ACSL)

ACSL Program Structure

```
PROGRAM
        INITIAL
                Statements executed before run begins
                State Variables do not contain initial conditions yet
        END ! Initial
        DYNAMIC
                DERIVATIVE
                        Statements to be integrated continuously
                        Statements are sorted.
                END ! Derivative
                DISCRETE
                        Statements to be executed at discrete points in time
                END ! DISCRETE
                        Statements to be executed each communication interval
        END ! Dynamic
        TERMINAL
                        Statements executed after the run terminates
        END !  Terminal
END ! Program
```

ACSL Program 1

Boost Converter Average-Value Model

ACSL File, Boostavg.csl

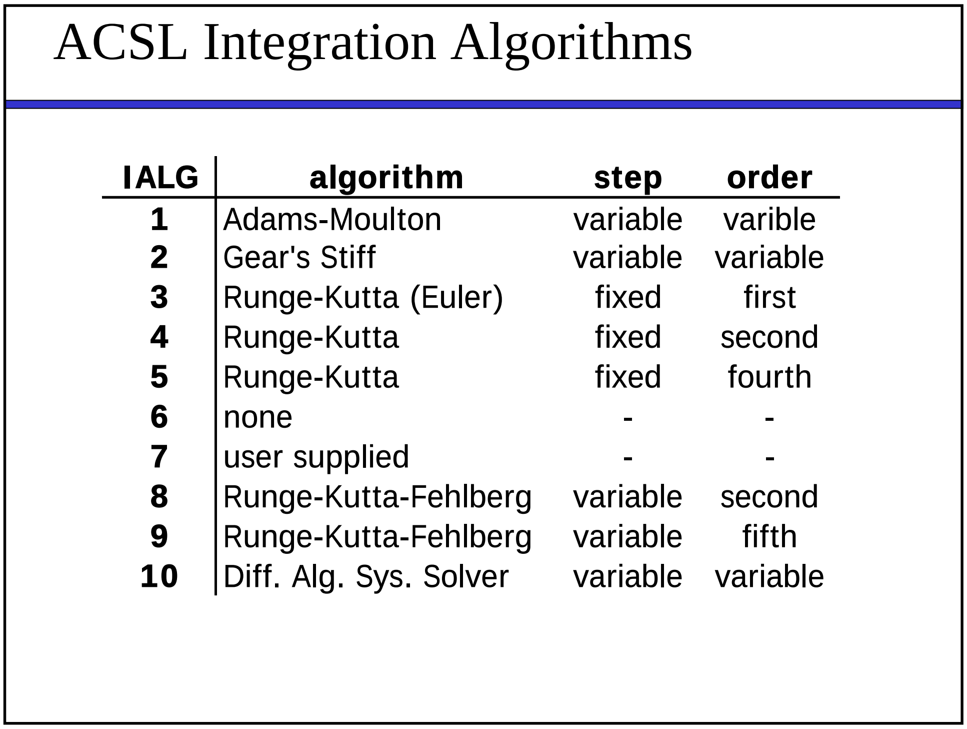
```
PROGRAM

DYNAMIC

ALGORITHM ialg=2
CINTERVAL cint=5.0e-4
MAXTERVAL maxt=2.0e-3
MINTERVAL mint=2.0e-5
NSTEPS nstep=1000
CONSTANT tstop=1.0
TERMT(t.ge.tstop,'Exit on Tstop')

DERIVATIVE

  CONSTANT r=60.0, l=4.48e-3, c=812.0e-6
  CONSTANT vi=300.0, vost=450.0
  CONSTANT voic=0.0, ilic=0.0

  INITIAL
    d = vi/vost
  END ! Initial

  v = d*vo
  i = d*il

  vl = vi-v
  ic = i-io

  pil = vl/l
  il = INTEG(pil,ilic)

  pvo = ic/c
  vo = INTEG(pvo,voic)

  io = vo/r

END ! Derivative

END ! Dynamic

END ! Program
```

Simulation Parameters

ALGORITHM - sets the integration algorithm
CINTERVAL - sets the rate that data is logged
MAXTERVAL - maximum integration interval
MINTERVAL - minimum integration interval
NSTEPS - number of integration intervals in one cint
TERMT - terminates program on logical condition
(t.GT.tstop) with message (Exit on Tstop)

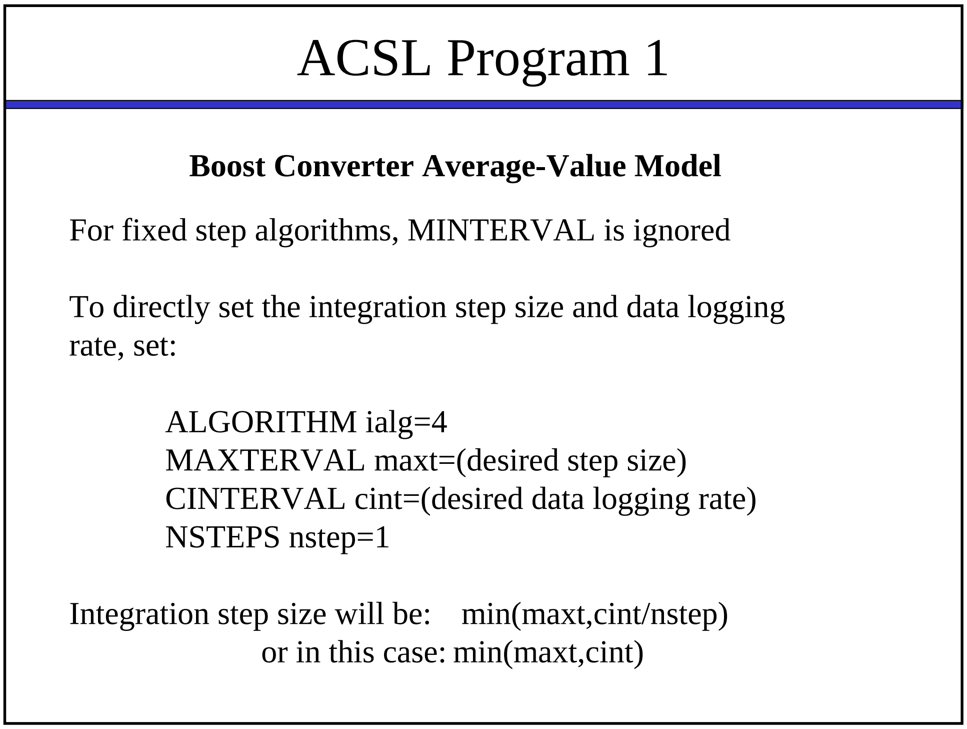
Other Statements

CONSTANTS - sets a variable as a constant
INTEG - integrates a variable, for example
x=INTEG(px,xic)
makes x equal to the time integral of px,
with an initial condition of xic



ACSL Integration Algorithms

IALG	algorithm	step	order
1	Adams-Moulton	variable	varible
2	Gear's Stiff	variable	variable
3	Runge-Kutta (Euler)	fixed	first
4	Runge-Kutta	fixed	second
5	Runge-Kutta	fixed	fourth
6	none	-	-
7	user supplied	-	-
8	Runge-Kutta-Fehlberg	variable	second
9	Runge-Kutta-Fehlberg	variable	fifth
10	Diff. Alg. Sys. Solver	variable	variable



ACSL Program 1

Boost Converter Average-Value Model

For fixed step algorithms, MINTERVAL is ignored

To directly set the integration step size and data logging rate, set:

```
ALGORITHM ialg=4
MAXTERVAL maxt=(desired step size)
CINTERVAL cint=(desired data logging rate)
NSTEPS nstep=1
```

Integration step size will be: min(maxt,cint/nstep)
or in this case: min(maxt,cint)

ACSL Program 1

Boost Converter Average-Value Model

For variable step algorithms, set nstep high

For example,

```
ALGORITHM ialg=2
CINTERVAL cint=5e-4
MAXTERVAL maxt=2e-3
MINTERVAL mint=2e-5
NSTEPS nstep=1000
```

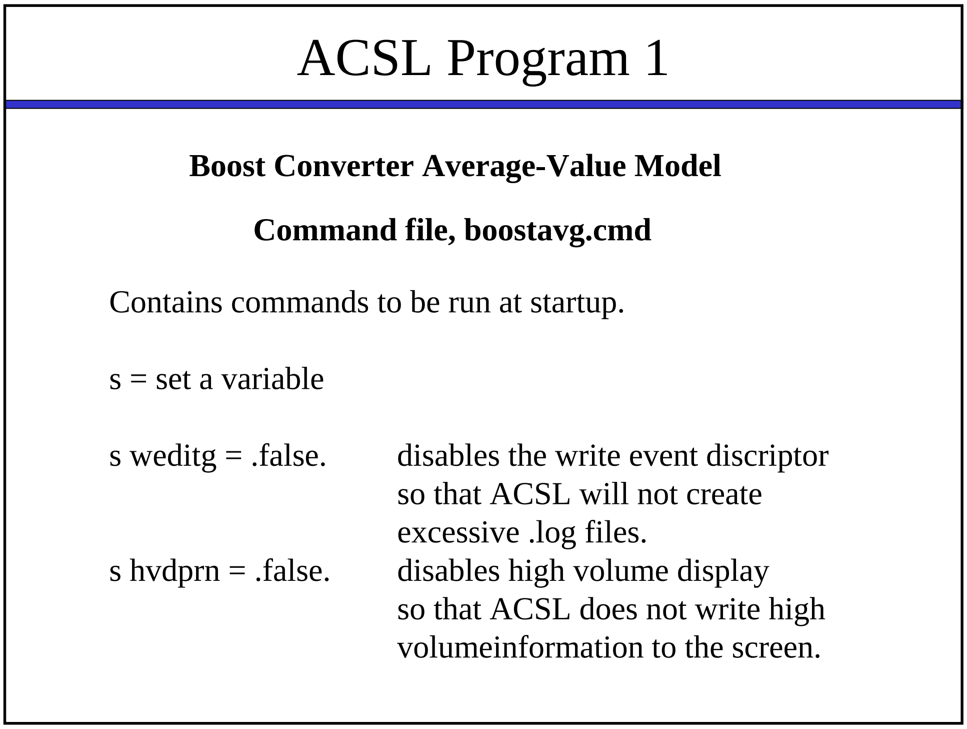
Integration step size starts at: cint/nstep
and is bounded by mint and maxt.
If nstep is large, step size will start out small.

ACSL Program 1

Boost Converter Average-Value Model

Command file, boostavg.cmd

```
s weditg=.f.
s hvdprn=.f.
s calplt=.f.
s strplt=.t.
s alcplt=.f.

procedure cal
   s strplt=.f.
   s calplt=.t.
   s alcplt=.t.
end

procedure str
   s calplt=.f.
   s strplt=.t.
   s alcplt=.f.
end

prepar t
prepar vo,il
```



ACSL Program 1

Boost Converter Average-Value Model

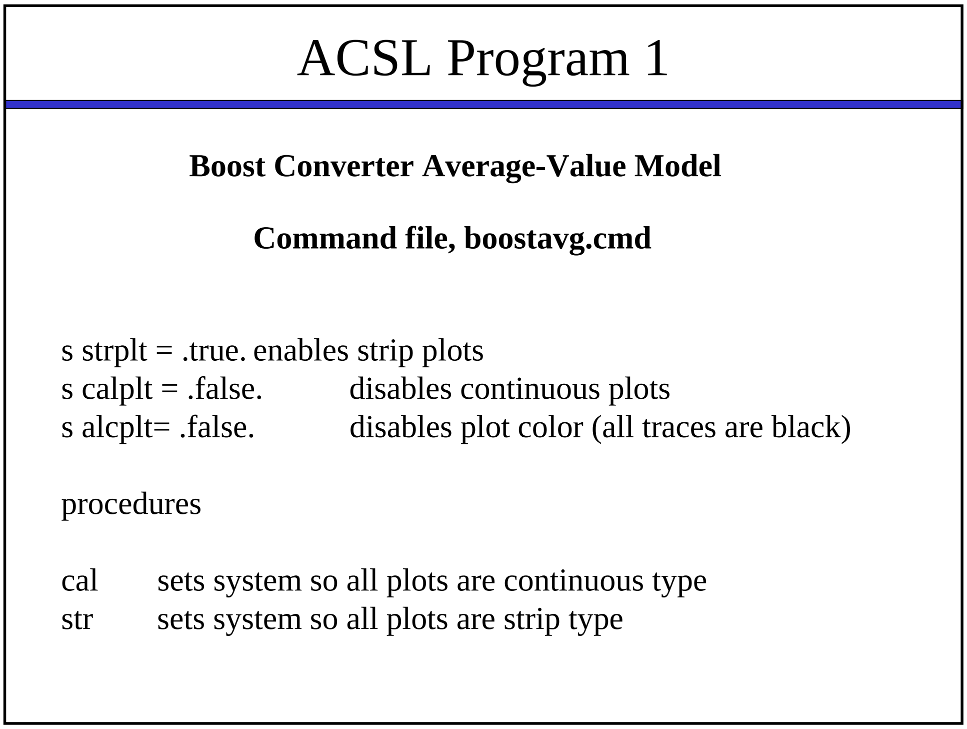
Command file, boostavg.cmd

Contains commands to be run at startup.

s = set a variable

s weditg = .false. disables the write event discriptor so that ACSL will not create excessive .log files.

s hvdprn = .false. disables high volume display so that ACSL does not write high volumeinformation to the screen.



ACSL Program 1

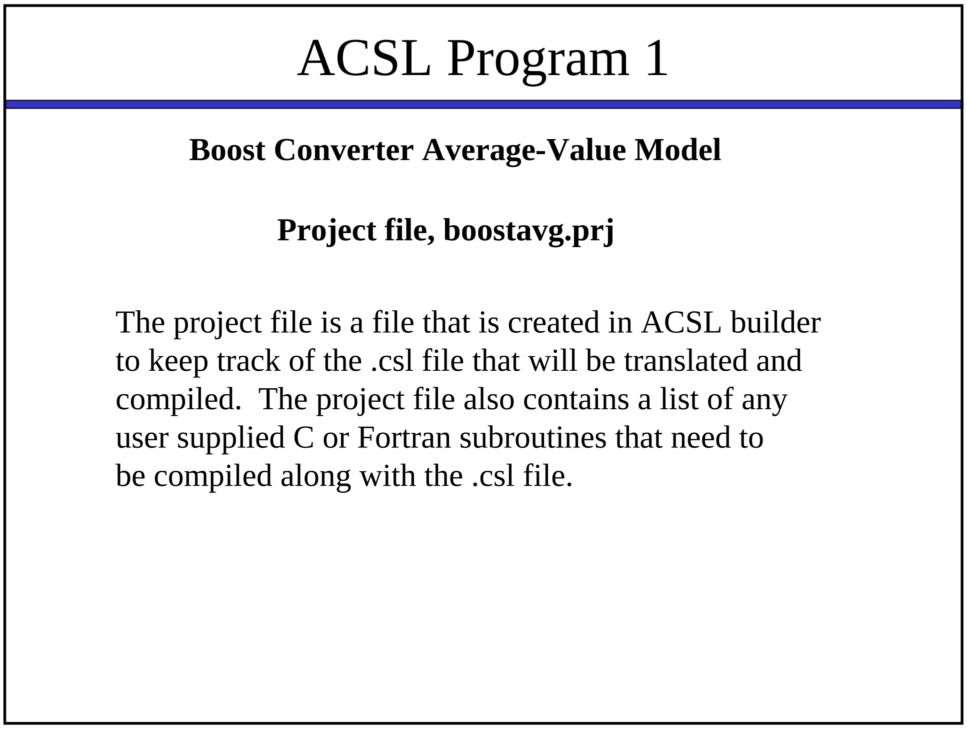
Boost Converter Average-Value Model

Command file, boostavg.cmd

s strplt = .true. enables strip plots
s calplt = .false. disables continuous plots
s alcplt= .false. disables plot color (all traces are black)

procedures

cal sets system so all plots are continuous type
str sets system so all plots are strip type

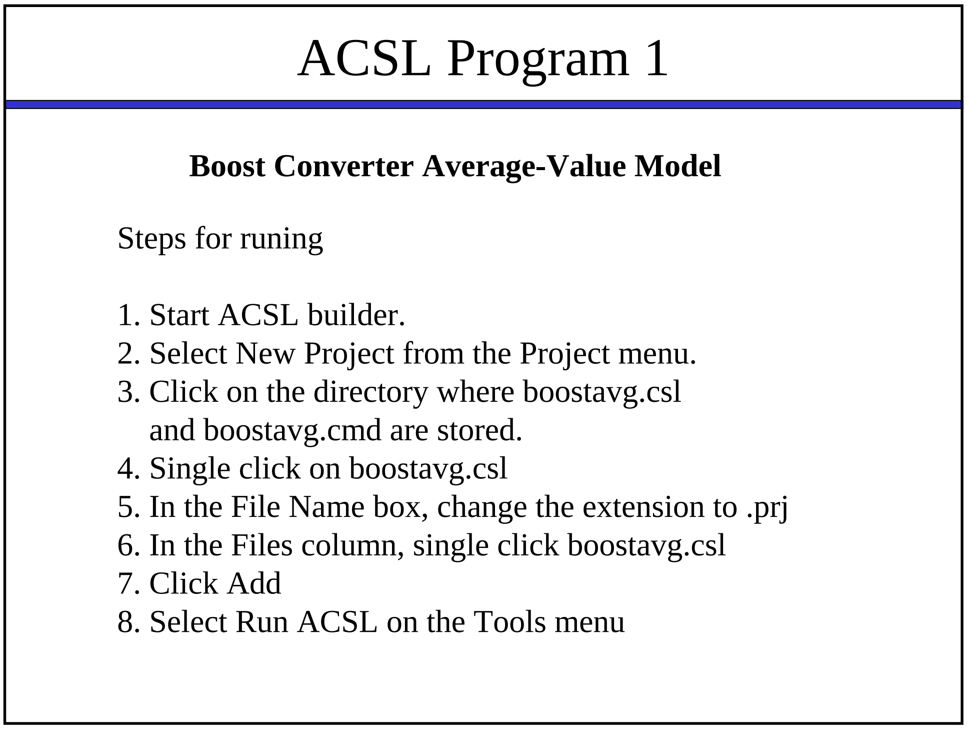


ACSL Program 1

Boost Converter Average-Value Model

Project file, boostavg.prj

The project file is a file that is created in ACSL builder to keep track of the .csl file that will be translated and compiled. The project file also contains a list of any user supplied C or Fortran subroutines that need to be compiled along with the .csl file.



ACSL Program 1

Boost Converter Average-Value Model

Steps for runing

1. Start ACSL builder.
2. Select New Project from the Project menu.
3. Click on the directory where boostavg.csl and boostavg.cmd are stored.
4. Single click on boostavg.csl
5. In the File Name box, change the extension to .prj
6. In the Files column, single click boostavg.csl
7. Click Add
8. Select Run ACSL on the Tools menu

ACSL Program 1

Boost Converter Average-Value Model

Once ACSL is running, type start and wait for the simulation to finish. To plot the output voltage and inductor current, type

```
plot vo,il/xlo=0/xhi=1
```

Print the results.

Type cal (see .cmd file for details) then type:

```
plot vo,il
```

Print the results

ACSL Program 1

Boost Converter Average-Value Model

Notes on running ACSL:

- To search for errors when compiling, ACSL creates a error file (.out). Perform a search on this file with the word error.
- You can change ialg, maxt, mint, cint, and nstps at the runtime command line (no need to quit ACSL and change the .csl file).
- For controlling the range on plots use /xlo, /xhi, /lo, and /hi
  - For example: plot vo/lo=0/hi=1000,il/lo=-200/hi=200/xl=0/xhi=0.0899
- To change the default x-axis variable use /xaxis
  - For example: plot vo/xaxis=il

ACSL Program 1

Boost Converter Average-Value Model

Notes on running ACSL:

- It helps to keep the notepad with the .csl file open, but minimized.
- If you change the .csl file, you must quit ACSL and re-start so that ACSL will re-compile.
- You must quit ACSL before you can run ACSL again. If not, you get a link error.
- If you change the .cmd file, quit ACSL, and re-start ACSL, the program will not recompile. It will simply issue the new commands.
- For future note: If you change a macro (.mac) file, save a new copy of the .csl file before running ACSL so it compiles properly.