DA - a program for computing differential approximants

Keith Briggs,
Department of Applied Mathematics,
University of Adelaide,
South Australia 5005.
kbriggs@maths.adelaide.edu.au

October 27, 2008

1 Introduction

This document describes the program DA. Its purpose is to compute differential approximants to series data. It has applications in statistical mechanics and other areas of applied mathematics. The theory and methods are as described in

A. J. Guttmann, Asymptotic Analysis of Power–Series Expansions, in C. Domb and J. L. Lebowitz (eds.) *Phase Transitions and Critical Phenomena*, volume **13**, pages 3-234 (1989).

The present program may be viewed as a development of the programs in the appendix to this article.

2 Theory

Differential approximants (DAs) generalize several series extrapolation and Padé approximant methods. The general mth order approximant is

$$\sum_{i=0}^{m} Q_i(z)\mathcal{D}^i f(z) = P(z) \tag{1}$$

where $\mathcal{D}^i = \frac{d^i}{dz^i}$, and $Q_i(z)$ and $P(z)$ are polynomials. The inhomogeneous term $P(z)$ may be absent. These polynomials are computed by DA in such a way that the series expansion of the solution $f(z)$ of the differential equation agrees with given series coefficients. The polynomial $Q_m(z)$ corresponding to the highest derivative always has constant term 1. Thus, zero is not a singular point. After

having computed a differential approximant, `DA` can estimate quantities of interest such as poles and exponents. Averaging of these quantities over a number of approximants is automatically performed. Approximants may be biased as described by Guttmann. Optionally, one may integrate the differential equation numerically to a desired endpoint.

3 Command file

The input to `DA` is a command file. Thus, the normal usage is

```
da < commandfile
```

Since `DA` reads standard input, it may also be used interactively. The command file may optionally contain series coefficients. Alternatively, the commands `file, id` and `read` allow one to read the series coefficients from another file. In this case, the series terms must be preceded by a unique identifying character string (on a line of its own). The series terms should follow, one per line, but any number of columns may be present. For example, the command `read 2 4` reads from the second column, four terms.

After inputting the series, several commands which transform the series are available. Then differential approximants may be fitted with a command of the form

$[n_m, \ldots, n_0; p]$

which fits mth order approximants with Q_j having degree n_j, and P (if present) having degree p. Each n_j $(j = 0, \ldots, m)$ may optionally be a range $n_{j\min}..n_{j\max}$, in which case all approximants in the range are computed. The inhomogeneous part $;p$ is optional. Occasionally, one finds that some approximants are defective and must be excluded from averaging. This is done with a command

$\sim[n_m, \ldots, n_0; p]$.

Ranges are not allowed here. Several commands may be combined on one line, separated with semicolons. The files `ex[1-4].da` contains examples of `DA` commands. The following tables summarize all commands.

4 Basic commands

keyword	arguments	action	note
`#`	*comment*	comment in command file	
`%`	*comment*	comment, echo to output	
`run`	*title*	new run, defines title, sets defaults	1
`data`	n m	read m terms in command file	2
		from column n, or until `end`	
`end`		end of series in command file	
`file`	*file*	opens series file	2
`id`	*id*	defines series identifier	
`read`	n m	read m terms from series file, column n	
`start`	n	ignore the first n series terms	
`skip`	m	skip m series terms between each term read	
`outf`	*title*	output series, floating point format	
`outi`	*title*	output series, integer format	
`[`$n_m, \ldots, n_0; p$`]`		fit mth order approximants	3
`~[`$n_m, \ldots, n_0; p$`]`		exclude this approximant	
`echo`	`on/off`	If `on`, commands are echoed to output	4
`shoa`	`on/off`	show/don't show approximants	4
`stop`		terminate job	

Notes Commands are either one per line, or are terminated with a `;`.

1. This command should be used at the start of each new analysis. It resets all options to their default values.

2. The commands `data` and `read` are alternative ways of inputting series. The former is more convenient for short series that can be embedded in the command file, the latter for longer series that already exist in another file. In this case the series must be preceeded by a unique identifying label (specified with the `id` command) and the file containing the series must have previously been opened with a `file` command. The series can be free format, one term per line. Any column can be read.

3. This is the command that starts the fitting process. Each n_j may be a range $n_{j\min}..n_{j\max}$. The inhomogeneous part *;p* is optional.

4. `off` is the default setting.

5 Advanced commands

keyword	arguments	action	note
Commands controlling averaging of results			
`all`	`on`	compute only approximants using all series terms	
`all`	`off`	compute every approximant	*
`band`	n ($n \geq 0$)	compute only approximants using at least all–n terms	1
`bias`	b n	bias to critical point at b, with multiplicity $n \leq 3$	2
`cp`	x_1 x_2	set acceptable critical point range	3
`ex`	e_1 e_2	set acceptable exponent range	3
`ep`	e_1 e_2	set acceptable range for integration endpoint	3
`gap`	`on/off*`	print/don't print distance to nearest pole	
Commands transforming the series			
`addc`	c	add c to constant term of series	
`subc`	c	subtract c from constant term of series	
`mulc`	c	multiply series by c	
`neg`		negate series	
`comc`	c	compose series with c	4
`log`		take logarithm of series	4
`difs`		differentiate series	4
`inte`		integrate series	4
`rec`		take reciprocal of series	4
`shif`	k	divide series by x^k	4
`delt`		take first difference of series	
`adds`	c	add c to all terms of series	
`logs`		take logarithm of series terms	
`exps`		take exponential of series terms	
Commands controlling integration			
`int`	`-1`	no integration	*
`int`	x ($x \geq 0$)	integrate to x, or...	5
`int1`	x_1 dt_1	integrate to x_1 in steps size dt_1, then	5
`int2`	x_2 dt_2	integrate to x_2 in steps size dt_2	5
`flag`	*flag*	insert *flag* on integration output	
Commands specific to the square–lattice Potts model			
`setq`	q	define q value for Potts series	6
`lotf`		get low T free energy from partition function	
`hitf`		get high T (free energy/T) from partition function	
`lotu`		get low T internal energy from partition function	
`hitu`		get high T internal energy from partition function	

Notes * default setting.

1. `band 0` is equivalent to `all`. On output, approximants that use all series terms are printed with the square brackets replaced by `*`.

2. This command switches on the computation of biassed approximants.

3. The inclusion of any of these three commands causes the average of results from all approximants which lie within the specified range.

4. These commands transform the series in the formal power series sense.

5. These commands switch on the integration of the differential approximant. `int1` and `int2` are used to study the region near the critical point with greater accuracy.

6. These commands were developed for a specific research project and are essentially undocumented. For a precise definition of the computation performed, see the file `da.m4`.

6 Acknowledgements

`DA` was written at Melbourne University as part of a project on series analysis of the Potts model funded by the Australian Research Council. Thanks are due to Professor Tony Guttmann, and to Dr. Thomas Prellberg who helped significantly with the implementation of biassed approximants. The differential equation solver was written by Professor Ernst Hairer, and the polynomial zero finder by Professor Bill Hager.

7 Installation

DA is written in m4 , not Fortran. This is so it can be automatically translated to either single, double or quadruple precision Fortran versions. At least double precision is recommended. All Unix systems possess m4 , though on some (such as Sun) it is essential to make sure the System V version is used (/usr/5bin/m4). For VAX and MSDOS use, obtain GNU m4 from the Free Software Foundation (prep.ai.mit.edu).

1. Make a directory ~/da. Put da.shar in it and unpack with

   ```
   sh da.shar
   ```

2. At the top of makefile choose one of the three precision options S,D or Q for single, double or quadruple precision. Note that not all Fortran compilers support quadruple precision.

3. Various compile–time maxima are set in approx.h. These are

   - maxk - maximum approximant order
   - maxm - maximum degree of coefficient polynomials
   - maxc - maximum series length

   Edit these if necessary.

4. make

5. Add the directory ~/da to your path.

6. To produce the documentation, do

   ```
   make doc
   ```

7. To remove object and .f files, you may

   ```
   make clean
   ```

8. The program may be tested with the command

   ```
   make test
   ```

The program is made up of the following files:

`makefile`	controls compilation and linking
`da.m4`	main program and user interface
`approx.m4`	differential approximant computation
`dlp.m4`	needed linpack routines
`odex.m4`	differential equation solver
`czero1.m4`	polynomial root finder
`realps.m4`	power series subroutines
`czindex.m4`	sort subroutine
`approx.h`	include file, sets compile–time maxima
`ex[1-4].da`	example command files
`da.tex`	this documentation
`verbatim.sty`	LaTeXstyle file needed for producing da.dvi

8 Example command files

```
% ex1.da -  K M Briggs, last revised 1993 May 3
% This example illustrates the estimation of the critical point
% from series data with small errors.
run Noisy series for 1/(1-x)
data 1 100
1.0001
1.0011
0.999
1.000004
1.00003
0.999
1.00001
0.999
0.999
1.00004
1.00003
1.0011
0.999
1.0004
1.003
1.0011
0.998
1.004
1.00003
end
cp 0.99 1.01; ex -1.1 -0.9  # look for CP 1, exponent -1
all
# Fit first-order DAs
[4..9,4..9]
stop
```

```
% ex2.da - K M Briggs, last revised 1993 May 3
% This example illustrates the use of inhomogeneous
% approximants to estimate critical points and exponents.
% The series enumerates staircase polygons on a square lattice,
% and was supplied by Prof. A J Guttmann.
run staircase polygons on a square lattice
data 1 100   # read terms from column 1
0
0
1
2
5
14
42
132
429
1430
4862
16796
58786
208012
742900
2674440
9694845
35357670
129644790
477638700
1767263190
6564120420
24466267020
91482563640
343059613650
4861946401452
18367353072152
69533550916004
263747951750360
end  # no more terms
cp 0.2499 0.2501 # CP is known to be 1/4
band 1           # use all or (all-1) series terms
shif 2           # remove factor x^2
# Fit first-order inhomogeneous DAs
[10..18,10..18;0..2]
stop
```

```
% ex3.da - K M Briggs, last revised 1993 May 3
% Example command file for illustrating biassing in DA
% test 1
run square lattice self-avoiding walks
% correct exponent is -1.34346
data 1 100
1.0
4.0
12.0
36.0
100.0
284.0
780.0
2172.0
5916.0
16268.0
44100.0
120292.0
324932.0
881500.0
2374444.0
6416596.0
17245332.0
46466676.0
124658732.0
335116620.0
897697164.0
2408806028.0
6444560484.0
17266613812.0
46146397316.0
123481354908.0
329712786220.0
881317491628.0
2351378582244.0
6279396229332.0
16741957935348.0
44673816630956.0
119034997913020.0
317406598267076.0
845279074648708.0
end
cp 0.379 0.38; bias 0.37905253 1; all
```

```
[8..9,8..9,8..9;-1..5]
#
% test 2
#
run (1-x)^(-3/2)*(1-x/2)^(3/2)+exp(-x)+(1-x)^(-5/4)
% Here we expect cp=1, ex=-3/2, cp=2, ex=3/2 and cp=1, ex=-5/4.
data 1 40
3.
1.
2.75000000000000000000000000000
2.28645833333333333333333333333
2.66959635416666666666666666666
2.77486979166666666666666666666
2.92533908420138888888888888888
3.05315394325861855515873015873
3.17360950121803889190890873015
3.28621040382923482797343474 42
3.39240980883544504747609402 55
3.49307743748696841967302731 86
3.58893084071262247183902547 40
3.68054497412179348039399088 52
3.76839130874997155675734047 30
3.85286170182385757057612474 48
3.93428589897448494039873959 76
4.01294437524027788073913857 43
4.08907796842621544884422909 12
4.16289522932686205946629657 64
4.23457811591932063068924263 30
4.30428646471581567709826539 26
4.37216154476437943773253919 28
4.43832891362811498111225153 98
4.50290073537810134401725645 66
4.56597767908535774802508172 78
4.62765048670535758546782793 02
4.68800127785933174070559945 04
4.74710464334623750461542345 29
4.80502856759566179328942545 50
4.86183521155108716323088953 79
4.91758158086056117408201768 76
4.97232009918878486998011664 08
5.02609910255246069564525437 37
5.07896326753201359427699831 90
5.13095398381798305007359021 66
```

```
5.18210967965514565769108776 63
5.23246610723700760397333873 23
5.28205659389160412772009363 26
5.33091226392141547395315942 59
end
run test 2, unbiased
all; gap on; cp 0.99 1.01;
[11..13,11..13,11..13;-1..4]
#
% test 3
#
run test 3, biased, no confluency
bias 1.0 1; all; gap on; cp 0.99 1.01;
[11..13,11..13,11..13;-1..4]
#
% test 4
#
run test 4, biased, confluency
bias 1.0 2; all; gap on; cp 0.99 1.01; ex -1.26 -1.24
[11..13,11..13,11..13;-1..4]
#
% test 5
#
run test 5, biased, confluency
bias 1.0 2; all; gap on; cp 0.99 1.01; ex -1.51 -1.49
[11..13,11..13,11..13;-1..4]
stop
```

```
% ex4.da - K M Briggs last revised 1993 May 5
% Reference: K M Briggs Math Comp 57, 435-439 (1992)
run Feigenbaum delta d=2 example
% Expected value 4.669210609
# Superstable parameters for period doubling of f(x)=a-x^2.
data 1 30
1.0
1.3107026413368328835635707 97
1.3815474844320614695406935 62
1.3969453597045606416724779 87
1.4002530812147827973250122 82
1.4009619629448410402961163 15
1.4011138049397761239008796 57
1.4011463258269461786472882 38
1.4011532908499238814746722 90
1.4011547825466178412186125 37
1.4011551020224639758940649 14
1.4011551704444112637653578 78
1.4011551850982972905309844 78
1.4011551882367109415437864 59
1.4011551889088630385632642 67
1.4011551890528174344922446 58
1.4011551890836480577946177 17
1.4011551890902510331817705 60
1.4011551890916651883071968 10
1.4011551890919680570294789 31
1.4011551890920329222350617 59
1.4011551890920468143757344 53
end
cp 4.66 4.7
#bias 4.669201907 1
bias 1 1
# Fit first-order DAs
[8..9,8..9;-1..3]



run Feigenbaum delta d=4 example
% Expected value 7.284686217
# Superstable parameters for period doubling of f(x)=a-x^4.
data 1 30
1.0
1.1457142071597121509311329693950263331663002050204 3
```

```
1.16526664006082979538370709992949739652314302792011
1.16793921291220414692907721354249606704722417485378
1.16830520105944963371922588221150092135389488995990
1.16835540907181676830282758563332140940365482312179
1.16836229983383860142950409189783868645969904160177
1.16836324569991458326517784057330913514803199869003
1.16836337554055553166228019046552190669273128766238
1.16836339336423916820558196682160680657697575725746
1.16836339581096846154000114749883566214043107528463
1.16836339614684129006083488518957747439495590225030
1.16836339619294798194300550815894244803990775357690
1.16836339619927724427393366292925823262091853934374
1.16836339620014608914509524287248588512753743235711
1.16836339620026535919219868642323541019600588290448
1.16836339620028173190111598998102329023634503791348
1.16836339620028397945281893704709414970649055218653
1.16836339620028428798384358036902389039007842373908
1.16836339620028433033721549537137167469554036618312
1.16836339620028433615124380800277589258667763681850
end
cp 7 8
bias 1 1
# Fit first-order DAs
[8..9,8..9;1..5]
stop
```