Convolution powers of characteristic functions

Praciano-Pereira, T. *

Sobral Matematica
Universidade Estadual Vale do Acaraú
March 5, 2012
tarcisio@member.ams.org
preprints da Sobral Matematica
no. 2012.01
Editor Tarcisio Praciano-Pereira
tarcisio@member.ams.org

Abstract

Este artigo apresenta um programa em `calc`, [1] que calcula a potência por convolução de uma função característica para um inteiro positivo arbitrário. O programa está sendo publicado sob a licença GPL e pode assim ser livremente usado ou modificado nos termos desta licença.

palavras chave:GPL, potências por convolução de função característica, programa em `Calc`, splines.

MSC: Primária 65D07, Secondária 65D15

This paper presents a `calc`, [1], program that calculates the convolution powers of the characteristic function of $[0, 1]$ for an arbitrary positive integer. The program is published under the GPL and can be freely used, modified in the terms of this license, GPL.

keywords:calc program, convolution power of characteristic function, GPL, splines.

MSC: Primary 65D07, Secondary 65D15

*Dep. de Computação - UeVA - Sobral tarcisio@member.ams.org

1 Introduction

This program is discussed thoroughly at the paper [2] but this author will happily discuss the program with people interested and showing potential to change the program with different aims.

This program is a long time work as it can be seen from [3], [4], [5] and the author is very happy to thanks his colleagues at Department of Computação of the university Vale do Acaraú for the leave which he is spending at the University of Aveiro which gave him the opportunity to finish this job as well as the Department of University of Aveiro, Portugal, for giving him a good visiting time.

There is a brief introduction to the program in the next section which this author suppose will be enough to run the program and test its capabilities and even make use of it to construct, for an instance, a base for a space of splines, which is the main idea under the program. Moreover the author suppose that the program is well documented with quite enough comments, so read the program before running it.

Other uses of this program may of pedagogical nature, for an instance to show the increase of oscillation of the derivatives of a function of compact support, existence of highly differentiable signals with compact support and even the construction of such signals with quite general assumptions. The graphic (1) shows a bad example of approximation with an interpolation projector con-

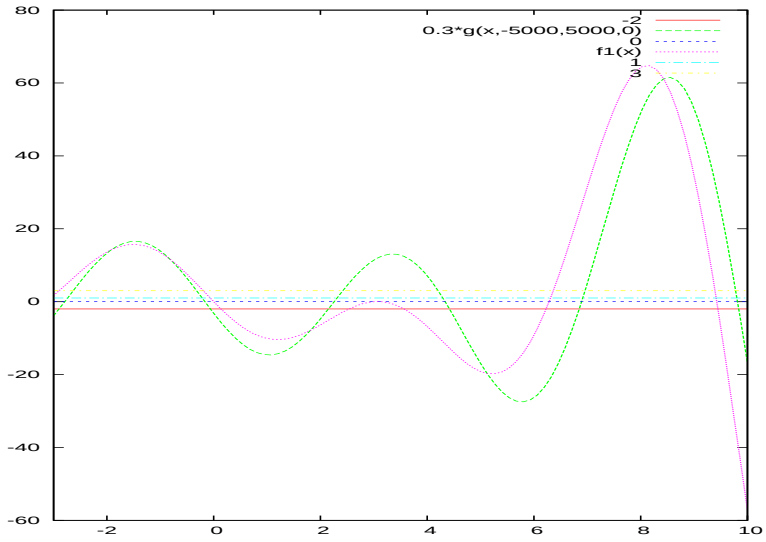

Figure 1: interpolation projector image of $f(x) = sin(x)(x-3)(x+5$

structed with translations of convolution power of a characteristic function. The *bad example* is a *good one* to show you that it is an approximation, with a little work something quite better can be done. See, [3], for something better done with an old method.

The program may be download of the link you can find at [5] it is copyleft under `GPL` and you can find in the program the meaning of this license and

where to obtain it.

2 A computer program

This is a program written in `Calc`, [1]. To have the graphics of $\chi^n_{[0,1]}$ for $n \geq 3$.

To run the program, download it from the link [5] and read it, it is text, so the code is provided to you with the program in the sense of `GPL`.

1. Start `Calc`, then you shall have a terminal of the calculator to make inputs. `Calc` is iterative, it is an *interpreted language*;

2. Supposing you have the code at the current directory named to

   ```
   convolution_powers.calc
   ```

   execute at the terminal of `Calc`:

   ```
   read "convolution_powers.calc"
   ```

   This will make `Calc` learn all the functions defined at

   `convolution_powers.calc`

   and you will be able to execute any of them. There is one function called `main()` which is written to do the job.

3. Execute "main(n)" selecting the desired value of n and stroke "enter" at the terminal of `Calc` and you will see f, f', f''.

   There is a problem of communication between `Calc` and `gnuplot`. `Calc` puts the symbol "~" in front of a big number which is not accurate to indicate that an approximation has been made. `gnuplot` will not understand this and the graphic will be wrong.

   Read the program to see how to undergo this problem.

4. If you want see the matrix, execute `imprime_matriz(a,n)` giving it a and n, but first run

   - a = cria_matriz(n); either repeat the the value of n or put a new one.
   - `imprime_matriz(a,n)` the parameter "a" has been created in the previous step, but you still need to put the same value of n as parameter. This indicates that the program is not well planned...Perhaps somebody will produce something better from this draft.

   alternatively run `main(n)` with the integer you want, and then `imprime_matriz(a,n)`.

5. Finally, you are welcome to contact the author for help with this program, and he will be happy to see a better version of the program, too.

3 The code

Observe that you can read this from the file at [5], with a text editor and even try it iteratively, you only need `Calc`, [1], and `gnuplot`, [6], and both are freely distributed under `GPL`.

```
#!/usr/bin/calc
## instructions at the end of the file
## this program should be executable under Linux
define copyleft()
{
printf("This program is distributed under  GPL. \n");
printf("If you do not know what is the GPL about, send and e-mail to the author \n");
printf("    tarcisio@member.ams.org \n, or find a version of GPL at www.fsg.org\n");
printf("but shortly this means that you can use freely the program or modify it\n");
printf("provided that this copyleft notice be maintained in any copy of modification\n");
printf("that you may do of the program as well as at the copies of it. \n");
printf("This the protection of the rights of the author that do not hinders\n");
printf("from a free use of the knowledge but that saves the memory of those that \n");
printf("have produced it.\n");
printf("This program has been written by  Tarcisio Praciano-Pereira \n");
printf("professor of  Universidade Estadual Vale do Acara - UeVA \n");
printf("Departamento de Computao/Ueva - Dep de Matemtica/www.ua.pt \n");
printf("http://www.sobralmatematica.org -  Sobral - Cear - Brasil\n");
printf("Aveiro, February - 2012 \n")
}
```

```
define  apeteco2()
{
   local temporario=0;
   printf("%s","============================================== \n");
   printf(" Hit a number and  <enter> to go ahead \n");
   scanf("%c",temporario);
}
```

```
define pow(x,n) {return(power(x,n));}
define bin(n,m) {if(n>=m) return(fact(n)/(fact(n-m)*fact(m)))
else return(bin(m,n));}
define Bin(n,m) {return( pow(-1,m)*bin(n,m));}
define Pow(x,n) {return(power(x,n)/fact(n));}
define cria_matriz(n){
local m,k,j, a=mat[n,n], soma;
m=0; k = 0; /* the line of order zero of a[i,j] */
while( k<n ){
```

```
a[m,k]=Bin(n-1,k);
k++;
}    /* * end of the line of order zero - this while() */
m=1; k=0; /* the line of order one of a[i,j] */
a[m,0] = a[0,0];
a[m,1] = a[m,0]*Pow(1,m);
k = 2;
while(k<n){
a[m,k]= a[m,k-1] + a[m-1,k-1]*Pow(1,m);
k++;
}                              /* end of this  while() */
m=2;k=0;   /* the other lines of  a[i,j] */
while(m<n) {
a[m,0] = a[0,0]; a[m,1] = a[m,0]*Pow(1,m);
k = 2;
while(k<n){
soma = a[m,k-1];  /* inicializes  soma */
j=1;
while(j<=m){
soma = soma + a[m-j,k-1]*Pow(1,j);
j++;
} /* end of the while producing one coefficient  */
a[m,k]=soma;      /* creates a[a,k]  */
k++;
} /** end of while producing one line */
m++; /* **** next line of matriz a[i,j]  */
} /* ***** end of  while producint a[i,j] for k >= 2  */
return(a);
} /* ******  end of  cria_matriz(n)  */

define imprime_matriz(a,n){   local i=0,j=0;
 while(i<n){
j=0;
while(j<n){
print a[i,j++],
}
print "\n";
i++;
}
}
## find the the interval where x lies and calculates f(x)
define f(x,a,n) {
local soma = 0, k=0, j=1, tr, m=n-1;
if ( x <=0 ) return(soma);
else if ( x <=1 ) {
soma = a[m,0]*Pow(x,m);
return(soma);
}
else if ( x < n ) {
tr=floor(x);
```

```
soma = a[m,tr];
j=1;
while(j<=m){
soma = soma + a[m-j,tr]*Pow(x-tr,j);
j++;
}
return(soma);
}
else return(soma);
}

## find the the interval where x lies and calculates f'(x)
define df(x,a,n) {
local soma = 0, k=0, j=1, tr, m=n-2;
if ( x <=0 ) return(soma);
else if ( x <=1 ) {
soma = a[m,0]*Pow(x,m);
return(soma);
}
else if ( x < n ) {
tr=floor(x);
soma = a[m,tr];
j=1;
while(j<=m){
soma = soma + a[m-j,tr]*Pow(x-tr,j);
j++;
}
return(soma);
}
else return(soma);
}

## find the the interval where x lies and calculates f''(x)
define d2f(x,a,n) {
local soma = 0, k=0, j=1, tr, m=n-3;
if ( x <=0 ) return(soma);
else if ( x <=1 ) {
soma = a[m,0]*Pow(x,m);
return(soma);
}
else if ( x < n ) {
tr=floor(x);
soma = a[m,tr];
j=1;
while(j<=m){
soma = soma + a[m-j,tr]*Pow(x-tr,j);
j++;
}
return(soma);
}
```

```
else return(soma);
}


/* creates the matrix of points for gnuplot
there will be an error of communication between calc
and gnuplot - calc put the sign ~ in front of large
numbers to indicate they are approximations - gnuplot
will not understand this. You have to edit the files
dados* to remove ~
 */
define cria_dados(a,n) {
local dadosf = fopen("dadosf", "w");
local dadosd1f= fopen("dadosd1f", "w");
local dadosd2f= fopen("dadosd2f", "w");
local x = -3, fim = n+5, delta = 0.01;
while(x < fim){
fprintf(dadosf, "%f %f \n %f %f \n", x,0, x, f(x,a,n));
fprintf(dadosd1f, "%f %f \n", x, df(x,a,n) ); /* derivative n */
fprintf(dadosd2f, "%f %f \n", x, d2f(x,a,n) ); /* derivative n */
x+=delta;
}
fclose(dadosf); fclose(dadosd1f); fclose(dadosd2f);
/* edit these files to remove '~' from big numbers */
}

/* this function creates the file "transfere" containing commands for
gnuplot. You can edit it to your needs and run gnuplot
with the command line "gnuplot transfere" - observe that
the files dados* have to bee edit too.
*/
define cria_transfere(n) { local leitura=2.3; /* to force a stop...*/
local inicio=-3, fim=n+3;  /* defines the range of the graphic  */
printf("Correct \"dados*\" erasing the signal \"~\" that \n");
printf( "calc puts in front of big numbers to indicate it is \n");
printf( "an approximate value - gnuplot do not understand this!  \n");
printf( "I will open \"dados*\" with joe! \n");
printf( "If you do not have \"joe\" this will produce and error\n");
printf( "but at this point all the job is done, stop the program\n");
printf( "edit the files \"dados*\" as explainded above and run\n");
printf( "gnuplot from the command line or do something else, for\n");
printf( "an instance run imprime_matriz(a,n) to see the matrix.\n");
printf( "Hit a number and <enter> to so I can verify you have read this....  \n");
scanf("%f",leitura);
system("joe dadosf dadosd1f dadosd2f");
local transfere = fopen("transfere", "w");
fprintf(transfere, "set xrange [%f:%f] \n", inicio, fim);
fprintf(transfere, "set title \" %d-th convolution power  \" \n",n);
fprintf(transfere, "set pointsize 0.1  \n");
fprintf(transfere, "plot -10,\"dadosf\", \"dadosd1f\", \"dadosd2f\" with points, 10 \n");
```

```
fprintf(transfere, "pause -2 \"Hit  enter to go ahead!\" \n");
fclose(transfere);
}
define main(n) {local a;
a=cria_matriz(n);
cria_dados(a,n);
cria_transfere(n);
system("gnuplot transfere");
}

printf("To run this program: \n");
printf("(1) Start calc \n");
printf("(2) At the terminal of calc execute: \n")
printf("read \"convolution_power.calc\"  \n" );
printf("execute the command: \n");
printf("main(n)  \n selecting the desired value for the \n convolution power. \n");
printf("This version of the program does the graphics of  f, f', f'' \n");
printf("and calculates the matrix of all coefficients for f and all its function-derivatives
##printf("Give-me the desired power n =  \n");
##n = 0;
##scanf("%d",n);
##main(n);
```

References

[1] David I. Bell, Landon Curt Noll and Ernest Bowen
CALC - arbitrary precision calculator
http://isthe.com/chongo/tech/comp/calc/

[2] Neves, A J M and Praciano-Pereira,T.
Convolution powers of a square signal
to appear (2012) - submitted

[3] Praciano-Pereira,T.
Splines por convolução
Preprints da Sobral Matematica - 2007.09
http://www.sobralmatematica.org/preprints

[4] Praciano-Pereira,T.
Convolução de funções caracterśticas em $\mathbf{R}^2$
Preprints da Sobral Matematica - 2011.05
http://www.sobralmatematica.org/preprints

[5] Praciano-Pereira,T.
convolution_power.calc a `calc` program to produce convolution powers of characteristic functions.

http://www.calculo-numerico.sobralmatematica.org/programas/convolution_power.calc

[6] Thomas Williams, Colin Kelley and many others
gnuplot, software to make graphics
http://www.gnuplot.info