

Appendix L. PASCAL include file output_calc.p

```

=====
PROCEDURE output_calc;

VAR
  i: INTEGER;
  factor: REAL;

Begin
  { calculate power consumption of mill }
  { evaluate fractional filling of voids in grinding media }
  IF (X[0]/Vo <= fines_voidage*fc)
    then U:= fc/(media_voidage*J)
    else U:= ((1-fines_voidage)*fc + X[0]/Vo)/(media_voidage*J);

  { evaluate density of pulp }
  IF x[0]+x[1]+x[2]<>0 then
    begin
      IF (X[0]/Vo <= fines_voidage*fc)
        then pulp_density:= (1-fines_voidage)*ore_density + X[0]/Vo
        else pulp_density:= ore_density*(X[1]+X[2])/(X[0]+X[1]+X[2])
          + X[0]/(X[0]+X[1]+X[2]);
      end
    else pulp_density:=0;

  { calculate power correction factor }
  power_correction:= 1 + media_voidage*pulp_density*U/ore_density;

  { evaluate fractional filling of mill }
  IF (U <= 1)
    then fractional_filling:= J
    else
      IF (X[0]/Vo <= fc)
        then fractional_filling:= (1-media_voidage)*J + fc
        else fractional_filling:= (1-media_voidage)*J
          + (1-fines_voidage)*fc + X[0]/Vo;

  { evaluate coarseness of load }
  IF (fc = 0) then coarseness:= max_coarseness;
  IF (J = 0) then coarseness:= min_coarseness;
  IF x[1]+x[2]<>0 then
    IF (fc <> 0) and (J <> 0) then
      coarseness:= (X[3] + X[4])/(X[1] + X[2])
    else coarseness:=max_coarseness;
  IF coarseness>max_coarseness then coarseness:=max_coarseness;
  coarseness:= (coarseness - min_coarseness)/
    (max_coarseness - min_coarseness);
  mx:= -(1 - 0.4/max_load)*coarseness + 1;
  my:= (power_max/0.32 - 1)*coarseness + 1;
  factor:= mx*fractional_filling;

  { calculate power consumption }
  IF (coarseness >= 0)
    then
      begin
        power:= Rose_sullivan(factor);
        power:= power_correction*Po*power + Pm;
        power:= Po*power*my;
      end
    else
      begin
        {power:= Pm;}
        writeln('coarseness < 0');
      end;

  { check fo overload condition }

```

```

{ check for maximum load level }
load_max_power:= max_load/mx;
IF (fractional_filling > load_max_power)
then begin
    FOR i:=3 to 4 do overload[i]:=1;
    {power:=Pm;}
    if opt_time then writeln('mill overfill');
end;

{ check for overfilling of media interstices }
IF (U <= 1) then
    FOR i:=1 to 4 do overload[i]:=1
else
IF (U > 1) and (U <= 1+overfill) then
    FOR i:=1 to 2 do begin
        overload[i]:= 1;
        {overload[i+2]:= 1 - reduced_wear;}
    end
else
begin
    FOR i:=1 to 4 do overload[i]:= 1;
    {power:=Pm;}
    if opt_time then writeln('interstices overfill');
end;

{ check for excess water dilution }
IF (X[1] + X[2] + X[3] + X[4] <> 0) then
IF (X[0]/(X[1] + X[2] + X[3] + X[4]) > water_critical) then
begin
    FOR i:=1 to 4 do overload[i]:= 1;
    {power:= Pm;}
    if opt_time then writeln('slipping ');
end
else
FOR i:=1 to 4 do overload[i]:= 1;

{ check for low load level }
IF (fractional_filling < minimum_load) then begin
    IF opt_time then writeln('reduced load ');
    FOR i:=1 to 2 do begin
        overload[i]:= 1;
        {overload[i+2]:= 1 + excess_wear;}
    end;
end;

end;

{=====

```

APPENDIX M. PASCAL PROGRAM PLOT.P

```

=====
const max_no_points=500;
      max_no_vars=30;

type strname=string[40];
      strpath=string[90];
      long_vector=array[1..2*max_no_points] of real;

var
  no_of_points:integer;
  datamatrix_x:array[1..max_no_vars] of long_vector;
  datamatrix_y:array[1..max_no_vars] of long_vector;
  file_var:text;
  file_group:text;
  group_name:strname;
  run_number:string[3];
  curve_no:integer;
  step_no:integer;
  path,path_group,path_select:strpath;

=====

```

```

{=====
PROGRAM plot (input,output);

$include '/usr/include/starbase.pl.h'$ { starbase constants and types }
$include '/users/bleloch/sim_dir/plot_var.p'$

CONST
    axis_height = 0.05;
    digits = 3;
    Edge = 1;
    EndOfLine = 0;

    header_height = 0.07;

TYPE
    data_points= array[0..2*max_no_points] of REAL;
    str30 = String[30];

VAR
    data_array: array [1..max_no_vars] of data_points;
    store_filename:array[1..max_no_vars] of string;
    dummy_name:string;
    name:array[1..max_no_vars] of string;
    device_type: CHAR;
    fildes: INTEGER;
    heading, x_label, y_label: str30;
    I, j: INTEGER;
    noofvars: INTEGER;
    num_points: array[1..max_no_vars] of INTEGER;
    String, filename: string[20];
    Vleft, Vright, Vbtm, Vtop: REAL;
    X: REAL;
    Xmax, Ymax, Xmin, Ymin, Zmax, Zmin: REAL;
    Xpower: INTEGER;
    Ya: REAL;
    color1_count,color2_count:integer;

$include '/usr/include/starbase.p2.h'$ { starbase external references }

{=====

Procedure initialise;
Begin
    Vleft:= 0.12;
    Vright:= 0.22;
    Vbtm:= 0.12;
    Vtop:= 0.10;          ( limits of the view port )
    Zmin:=0;
    Zmax:=1;
    write('Enter the title for the plot: ');
    readln(heading);
    { write('Enter the X-axis label: ');
    readln(x_label);)
    x_label:='Time Units';
    { write('Enter the Y-axis label: ');
    readln(y_label);)
    y_label:='Response';
End;

{=====

PROCEDURE Tics (fildes: INTEGER; Axis: char;
                Location, Start, Finish, Delta, Size: REAL;
                Major: INTEGER);

VAR
    Tic:    INTEGER;

```

```

TicMks: array[0..199] of REAL; {200 tics maximum}
Begin
  IF (Axis='x') or (Axis='X')
  then
    begin
      X:= Start;
      Tic:=0;
      I:=0;
      WHILE X <= Finish do
        begin
          TicMks[I ]:=X;
          TicMks[I+1]:=Location+Size*(1+ord((Tic MOD Major)=0));
          TicMks[I+2]:=0.0;
          TicMks[I+3]:=X;
          TicMks[I+4]:=Location+Size*(1+ord((Tic MOD Major)=0));
          TicMks[I+5]:=1.0;          {draw there}
          X:=X+Delta;
          Tic:=Tic+1;
          I:=I+6;
        end;
      end;
    IF (Axis='y') or (Axis='Y')
    then
      begin
        Ya:= Start;
        Tic:=0;
        I:=0;
        WHILE Ya <= Finish do
          begin
            TicMks[I ]:=Location+Size*(1+ord((Tic MOD Major)=0));
            TicMks[I+1]:=Ya;
            TicMks[I+2]:=0.0;
            TicMks[I+3]:=Location+Size*(1+ord((Tic MOD Major)=0));
            TicMks[I+4]:=Ya;
            TicMks[I+5]:=1.0;          {draw there}
            Ya:=Ya+Delta;
            Tic:=Tic+1;
            I:=I+6;
          end;
        end;
        polyline2d(filJes, TicMks, I DIV 3, 1);
      End; {procedure Tics }

```

```

=====
procedure find_limits;
var i,j:integer;
    x,y:real;
begin
  Xmax:= -maxint;
  Ymax:= -maxint;
  Xmin:= maxint;
  Ymin:= maxint;
  for i:=1 to noofvars do begin
    path:=path_select;
    strappend(path,store_filename[i]);
    reset(file_var,path);
    readln(file_var,dummy_name);
    readln(file_var,no_of_points);
    for j:=1 to no_of_points do begin
      readln(file_var,x,y);
      if x>Xmax then Xmax:=x;
      if x<Xmin then Xmin:=x;
      if y>Ymax then Ymax:=y;
      if y<Ymin then Ymin:=y;
    end;
    Ymax:=Ymax+(Ymax*0.02); { spacing }

```

```

        Ymin:=Ymin/(\min*0.02);
        close(file_var);
    end;
end;
{=====}

PROCEDURE read_data;
VAR
    i, j: INTEGER;
Begin
    i:= 1;
    write('What run number? ');readln(run_number);
    path:='/users/bleloch/sim_dir/plot_dir_';
    strappend(path,run_number);path:=strtrim(path);
    strappend(path,'/group_dir/');
    write('Name of group to be plotted? ');readln(group_name);
    strappend(path,group_name);
    path:=strtrim(path);
    reset(file_group,path);
    readln(file_group,path_select);
    i:=0;
    while not eof(file_group) do begin
        i:=i+1;
        readln(file_group,store_filename[i]);
    end; { while }
    noofvars:=i;
    find_limits;
    { convert data for polyline2d }
    for i:=1 to noofvars do
        begin
            path:=path_select;
            strappend(path,store_filename[i]);
            reset(file_var,path);
            readln(file_var,name[i]);
            writeln('reading ',name[i]);
            readln(file_var,no_of_points);
            num_points[i]:=no_of_points;
            j:=0;
            repeat
                readln(file_var,data_array[i,j],data_array[i,j+1]);
                j:=j+2;
            until j>=(2*no_of_points-2);
        end;
    end;
end;
{=====}

PROCEDURE append_label (VAR axislabel: Str30; VAR exp: INTEGER);
VAR
    dummy          :INTEGER;
    exp_str         :String[10];
    label_length    :INTEGER;
    power           :INTEGER;
Begin
    IF (abs(Xmax) > 999) or (abs(Xmax) < 100)
    then
        begin
            power:= trunc(ln(abs(Xmax))/ln(10));
            exp:= power - digits + 1;
            label_length:= strlen(axislabel)+1;
            IF (exp < 0)
            then
                IF (abs(exp) < 10)
                then strwrite(axislabel,label_length,dummy,' xE-0',abs(exp):0)
                else strwrite(axislabel,label_length,dummy,' xE-',abs(exp):0)
            end;
        end;
    end;
end;

```

```

else
  IF (abs(exp) < 10)
    then strwrite(axislabel,label_length,dummy,' xE+0',abs(exp):0)
    else strwrite(axislabel,label_length,dummy,' xE+',abs(exp):0);
  end; { IF }
End; { procedure append_label }

{=====}

Begin { main program }
{ clear any previous plots from the screen }
{ Begin reading specifications of the plot }
REPEAT
  write('Do you want the output to go to the Screen or the Plotter');
  write(' (S/P) ? ');
  readln(device_type);
  CASE device_type of
    'S','s': Begin
      fildes:= gopen('/dev/crt', OUTDEV, 'hp300h', INIT);
      device_type:= 'S';
      clear_view_surface(fildes);
    End;
    'P','p': Begin
      fildes:= gopen('/dev/plot', OUTDEV, 'hppl', INIT);
      device_type:= 'P';
    End;
    otherwise writeln('Invalid option entered');
  End; { case }
  UNTIL (device_type='S') or (device_type='P');
  IF fildes > 0 then Begin
    clear_view_surface(fildes);
    initialise;
    read_data;
    mapping_mode(fildes, DISTORT);
    { write the graph heading }
    text_alignment(fildes, TA_CENTER, TA_CAP, 0.0, 0.0);
    character_height(fildes, header_height);
    CASE device_type Of
      'S': begin
        X:= 0.4985;
        WHILE X <= 0.5015 do begin
          text2d(fildes, X, 1.0, heading, VDC_TEXT, EndOfLine);
          X:=X + 0.0005;
        End;
      end;
      'P': begin
        X:= 0.4995;
        WHILE X <= 0.5005 do begin
          text2d(fildes, X, 1.0, heading, VDC_TEXT, EndOfLine);
          X:= X+0.0005;
        End;
      end;
    End; { case }

    { write the x axis label ----- }
    text_alignment(fildes, TA_CENTER, TA_BASE, 0.0, 0.0);
    character_height(fildes, axis_height);
    append_label(x_label, Xpower);
    text2d(fildes, 0.5, axis_height/2, x_label, VDC_TEXT, EndOfLine);
    text_alignment(fildes,TA_RIGHT,TA_TOP,0.0,0.0);
    character_height(fildes,0.02);
    color1_count:=1;color2_count:=1;
    for j:=1 to noofvars do begin
      if name[j,1]='x' then begin
        text_color_index(fildes,color1_count);
        color1_count:=color1_count+1;

```

```

                                text2d(filides,0.9,0.8-j/40,name[j],VDC_TEXT,0)
                                end
                        else begin
                                text_color_index(filides,color2_count);
                                color2_count:=color2_count+1;
                                text2d(filides,0.9,0.8-j/40,name[j],VDC_TEXT,0);
                                end;
                        end;
                        text_color_index(filides,1);
{ write the y axis label ----- }
text_alignment(filides, TA_CENTER, TA_CAP, 0.0, 0.0);
character_height(filides, axis_height);
text_orientation2d(filides, -1.0, 0.0, 0.0, 1.0);
text2d(filides, 0.0, 0.5, y_label, VDC_TEXT, EndOfLine);
vdc_extent(filides,
            Xmin - (Xmax - Xmin) / (1 - Vleft - Vright) * Vleft,
            Ymin - (Ymax - Ymin) / (1 - Vtop - Vbtm) * Vbtm,
            0.0, { Zmin unused },
            Xmax + (Xmax - Xmin) / (1 - Vleft - Vright) * Vright,
            Ymax + (Ymax - Ymin) / (1 - Vtop - Vbtm) * Vtop,
            0.0); { Zmax unused }
clip_rectangle(filides,xmin,xmax,ymin,ymax);
interior_style(filides, INT_HOLLOW, Edge);
rectangle(filides, Xmin, Ymin, Xmax, Ymax);

{ plot the tics ----- }
Tics(filides, 'X', Ymin, Xmin, Xmax, (Xmax-Xmin)/100, (Ymax-Ymin)*0.01, 5)
Tics(filides, 'X', Ymax, Xmin, Xmax, (Xmax-Xmin)/100, (Ymax-Ymin)*0.01, 5)
Tics(filides, 'Y', Xmin, Ymin, Ymax, (Ymax-Ymin)/40, (Xmax-Xmin)*0.005, 5)
Tics(filides, 'Y', Xmax, Ymin, Ymax, (Ymax-Ymin)/40, (Xmax-Xmin)*0.005, 5)

{ plot the data ----- }
color1_count:=1;
color2_count:=1;
FOR j:=1 to noofvars do begin
        If store_fil_name[j,1]='s' then begin { ie is a state }
                line_color_index(filides,color1_count);
                line_type(filides,1);
                polyline2d(filides, data_array[j], num_points[j]-2,0)
                color1_count:=color1_count+1;
        end
        else begin
                line_color_index(filides,color2_count);
                line_type(filides,0);
                polyline2d(filides, data_array[j], num_points[j]-2,0);
                color2_count:=color2_count+1;
        end;
end;
line_color_index(filides, 1);

{ label the X tic marks ----- }
clip_indicator(filides, CLIP_TU_VDC);
text_orientation2d(filides, 0.0, 1.0, 1.0, 0.0); {back to default}
character_height(filides, 0.02*(Ymax-Ymin));
text_alignment(filides, TA_CENTER, TA_CAP, 0.0, 0.0);
Ya:= Ymin-(Ymax-Ymin)*0.01;
X:=Xmin;
WHILE X <= Xmax do
begin
        strwrite(Strng, 1, I, round(X/10*(Xpower*ln(10))):0);
        text2d(filides, X, Ya, Strng, VDC_TEXT, EndOfLine);
        X:= X+(Xmax-Xmin)/10;
end;

{ label the Y tic marks ----- }
text_alignment(filides, TA_RIGHT, TA_HALF, 0.0, 0.0);

```

```

X:=Xmin-(Xmax-Xmin)*0.005;
Ya:=Ymin;
WHILE Ya <= Ymax do
  begin
    strwrite(Strng, 1, I, Ya:6:2);
    text2d(fildes, X, Ya, Strng, VDC_TEXT, EndOfLine);
    Ya:=Ya+(Ymax-Ymin)/8;
  end;
  fildes:= gclose (fildes);
End { IF fildes > 0 then ... }
else writein ('Starbase unable to open correctly: ERROR# ',fildes);
End. { main program }

```

```

{=====}

```

APPENDIX N. VARIABLES DUMPED DURING A SIMULATION RUN

Autogenous Grinding Circuit Simulation

Setup file number? 6
 Start time 0.00
 Stop time 800.00
 Logging interval 4.00
 File containing initial conditions: init_1
 File containing plant constants: const_6
 Optimization start time= 50.00
 Estimator sampling time 0.50
 Optimization update time= 25.00
 Disturbance start time 400.00
 Magnitude of change 0.20
 Estimation stop time 800.00
 Magnitude of feed_rate excitation signal 1.00
 Magnitude of mill_water excitation 1.00
 Estimation only FALSE
 Covariance resetting on: TRUE

Running Simulation

{-----}

Time 25.0

Model Variables

Inputs

solids feedrate 4.44
 mill water 5.00

Outputs

Power draft 0.84

Inside mill

fractional filling of mill 0.41
 fraction of grinding media 0.41
 fractional filling of interstices 0.74

States

x[0]= 11.47
 x[1]= 5.90
 x[2]= 8.16
 x[3]= 31.47
 x[4]= 20.26
 x[5]= 31.52
 x[6]= 8.37
 x[7]=10.21

Regression vector:

0.841 0.845
 4.446 4.442
 5.006 5.000
 1.000

Parameters:

1.455-0.498
 0.012-0.122
 -0.056 0.124
 0.187 0.000

P_mat:

273.34 236.38 383.54
 376.04 311.13 304.83 1.00

Gradients 0.000 0.000
 Estimation error -0.0015

{-----}

Time 200.0

Model Variables

Inputs

solids feedrate 4.10
mill water 5.03

Outputs

Power draft 0.84

Inside mill

fractional filling of mill 0.52
fraction of grinding media 0.52
fractional filling of interstices 0.72

States

x[0]= 10.04
x[1]= 7.48
x[2]= 11.63
x[3]= 46.63
x[4]= 19.30
x[5]= 24.27
x[6]= 10.05
x[7]= 15.64

Regression vector:

0.840 0.840
4.104 4.109
5.037 5.039
1.000

Parameters:

1.461-0.570
0.015-0.023
-0.020 0.016
0.146 0.000

P_mat:

53.30 53.27 34.80
33.31 30.36 29.51 52.09

Gradients -0.108-0.025
Estimation error -0.0005

{-----}
Time 400.0

Model Variables

Inputs

solids feedrate 3.96
mill water 4.91

Outputs

Power draft 0.84

Inside mill

fractional filling of mill 0.50
fraction of grinding media 0.50
fractional filling of interstices 0.72

States

x[0]= 10.02
x[1]= 7.42
x[2]= 11.05
x[3]= 45.44
x[4]= 18.69
x[5]= 24.81
x[6]= 10.13
x[7]= 15.14

Regression vector:

0.841 0.841

3.959 3.976
4.908 4.910
1.000

Parameters:

1.443-0.588
0.015-0.014
-0.012 0.011
0.125 0.000

P_mat:

47.31 47.25 31.08
28.02 26.05 29.21 43.69

Gradients 0.008 -0.017
Estimation error 0.0001

<----->
Time 450.0

Model Variables

Inputs

solids feedrate 4.28
mill water 5.17

Outputs

Power draft 0.89

Inside mill

fractional filling of mill 0.02
fraction of grinding media 0.02
fractional filling of interstices 0.59

States

x[0]= 10.26
x[1]= 7.17
x[2]= 11.54
x[3]= 55.12
x[4]= 23.93
x[5]= 24.82
x[6]= 9.72
x[7]= 15.44

Regression vector:

0.893 0.894
4.287 4.287
5.178 5.180
1.000

Parameters:

1.486-0.544
0.021-0.011
-0.018-0.003
0.116 0.000

P_mat:

36.22 36.60 31.82
30.85 26.26 22.79 47.70

Gradients 0.429-0.640
Estimation error -0.0004

<----->
Time 600.0

Model Variables

Inputs

solids feedrate 4.12
mill water 4.90

Outputs

Power draft 0.86
 Inside mill
 fractional filling of mill 0.62
 fraction of grinding media 0.62
 fractional filling of interstices 0.63

States
 x[0]= 9.74
 x[1]= 7.43
 x[2]= 12.65
 x[3]= 55.96
 x[4]= 23.24
 x[5]= 23.40
 x[6]= 9.00
 x[7]= 16.68

Regression vector:
 0.855 0.855
 4.112 4.115
 4.894 4.894
 1.000

Parameters:
 1.473-0.557
 0.013-0.013
 -0.007 0.004
 0.088 0.000

P_mat:
 52.96 52.91 37.64
 35.26 31.35 33.97 50.51

Gradients 0.001-0.052
 Estimation error -0.0003

{-----}
 Time 800.0

Model Variables

Inputs

 solids feedrate 3.96
 mill water 4.81

Outputs

 Power draft 0.87

Inside mill

 fractional filling of mill 0.61
 fraction of grinding media 0.61
 fractional filling of interstices 0.63

States

x[0]= 9.73
 x[1]= 7.46
 x[2]= 12.06
 x[3]= 55.02
 x[4]= 22.51
 x[5]= 23.82
 x[6]= 9.95
 x[7]= 16.20

Regression vector:
 0.867 0.867
 3.953 3.970
 4.801 4.796
 1.000

Parameters:
 1.471-0.560
 0.011-0.008
 -0.005 0.001

0.084 0.000

P_mat:

49.03	49.21	30.62	
28.79	28.35	25.47	45.83

Gradients 0.003-0.020
Estimation error 0.0002

{-----}

Author Bleloch Mark John

Name of thesis An Adaptive Optimizing Regulator For An Autogenous Mill. 1987

PUBLISHER:

University of the Witwatersrand, Johannesburg

©2013

LEGAL NOTICES:

Copyright Notice: All materials on the University of the Witwatersrand, Johannesburg Library website are protected by South African copyright law and may not be distributed, transmitted, displayed, or otherwise published in any format, without the prior written permission of the copyright owner.

Disclaimer and Terms of Use: Provided that you maintain all copyright and other notices contained therein, you may download material (one machine readable copy and one print copy per page) for your personal and/or educational non-commercial use only.

The University of the Witwatersrand, Johannesburg, is not responsible for any errors or omissions and excludes any and all liability for any errors in or omissions from the information on the Library website.