```
// Hydrodynamic Journal Bearing, iterative solution with Scilab
// Allowing for variation of oil viscosity with temperature
// 13th February 2015 - David J Grieve
printf(" \n");
printf("Analysis of a journal bearing with hydrodynamic lubrication, \n\n");
printf("To provide data about the variation of viscosity (Pa.s) with temperature, deg. C following is needed:,\n");
printf("Programme calculates viscosity in Pa.s using this formula: mU = (mu0*exp(b/(TC(i)+pluscon)))/1000.0 \n");
printf("Data is for straight SAE grades 10 20 30 or 60 \n");
printf("For grade 10: mu0=0.1975  b=468  pluscon=53 \n");
printf("For grade 20: mu0=0.17  b=520  pluscon=53 \n");
printf("For grade 30: mu0=0.1974  b=520  pluscon=53 \n");
printf("For grade 40: mu0=0.1573  b=610  pluscon=53 \n");
printf("For grade 50: mu0=0.187  b=650  pluscon=53 \n");
printf("For grade 60: mu0=0.23375  b=660  pluscon=53 \n");// Set up frame for input data
// The sample data is a test example for comparison  see related web page
labels=["Oil in Temp. deg. C";"mu0";"b";"pluscon";"Journal diam. mm";"Journal length, mm";"Radial clearance, mm";"Load, N";"Shaft RPM";..
"Number of intervals round journal";"Number of intervals along bearing length";..
"SoR factor to be used, less than 1";"Number of iterations";"Eccentricity ratio";"Oil density, kg/m3, about 850";..
"Oil specific heat, J/kg.deg.C, about 1800";"Oil supply pressure, Pa";"Oil supply hole diameter, mm"];

[ok,startTemp,mu0,b,pluscon,JDiam,JLength,radialClear,JLoad,shaftRPM,circInts,widthInts,sOR,iTeration
getvalue("Enter the fifteen items of data listed below",labels,list("vec",1,"vec",1,"vec",1,"vec",1,"vec",1,..

"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec",1,"vec"

["60";"0.1974";"550";"53";"40";"40";"0.04";"2500";"1800";"180";"16";"0.9";"50";"0.59";"850";"1800";"1
printf("Input data: \n\n");
printf("Oil in temperature deg C: %f\n",startTemp);
printf("mu0: %f\n",mu0);
printf("Constant b: %f\n",b);
printf("Constaaant pluscon: %f\n",pluscon);
printf("Journal diameter, mm: %f\n",JDiam);
printf("Journal Length, mm: %f\n",JLength);
printf("Radial clearance, mm: %f\n",radialClear);
printf("Load, N: %f\n",JLoad);
printf("Shaft RPM: %f\n",shaftRPM);
// printf("Lubricant viscosity, Pas: %f\n",lubeVisc);
printf("Number of intervals round full circumference of journal: %f\n",circInts);
printf("Number of intervals along length of journal: %f\n",widthInts);
printf("SoR factor to be used: (usually less than 1): %f\n",sOR);
printf("Number of iterations to be used: %f\n",iTerations);
printf("Eccentricity ratio: %f\n",eccen); //less than 1
printf("Oil density, kg/m3,(about 850): %f\n",oilDen);
printf("Specific heat of oil, J/kg.degC, (about 1800): %f\n",oilSpecHeat);
printf("Oil supply pressure, Pa: %f\n",oilSupPress);
printf("Oil hole diameter, mm: %f\n",oilHoleDia);
// Next - Start calculations;
// Calculate start viscosity from values input
mU=1:1:circInts;
mU(1)=(mu0*exp(b/(startTemp+pluscon)))/1000.0; // Using modified eqn from: A S Seireg and S Dandage, ref 1.
mU(2)=mU(1);
// Calculate Bearing characteristic number or Sommerfeld number;

somNum=(JDiam/(radialClear*2.0))*(JDiam/(radialClear*2.0))*mU(1)*shaftRPM*JDiam*JLength/(60.0*JLoad*1
```

```
0051  printf("    \n");
0052  printf("Calculation results: \n\n");
0053  printf("Oil viscosity at input temp. Pa.s: %f\n",mU(1));
0054  printf("Bearing characteristic number or Sommerfeld number: %f\n",somNum);
0055  avProjPress=JLoad/(JDiam*JLength);
0056  printf("Average projected pressure MPa: %f\n",avProjPress);
0057  dx=JDiam*3.14159/(1000.0*circInts);
0058  dx2=dx*dx;
0059  dy=JLength/(1000.0*widthInts);
0060  dy2=dy*dy;
0061  dxdy=dx*dy;
0062  dx2dy2=2*(dx*dx)+2*(dy*dy);
0063  dTheta=2*3.14159/circInts; //dTheta is in radians
0064  vel=3.14159*JDiam*shaftRPM/(60.0*1000.0);
0065  // Next - Theoretical attitude angle
0066  // theoAt=180.0*atan(sqrt(1-eccen*eccen)/eccen)/4.0;
0067  // printf("Theoretical attitude angle, degrees: %f\n",theoAt);
0068  // Next calc film thicknesses
0069  x1=1;
0070  for i=1:circInts,h(i:i,1:widthInts+1)=radialClear*(1+eccen*cos(i*dTheta))/1000.0, end;
0071  //for i=1:circInts, Specifies a point at 'top' of an element in a column, round
 bearing
0072  //for j=1:widthInts, Specifies point at the 'left' of an element in a row along the
 length of the bearing
0073  shearForce=0;
0074  for i=1:circInts,
0075  fric=vel*mU(1)*JLength*JDiam*dTheta/(h(i:i,1:1)*2000000.0),
0076  shearForce=shearForce+fric,
0077  end;
0078  // Use h values to determine the mass of oil in each longitudinal slice along journal
0079  mSum=0;
0080  for i=1:circInts-1,
0081  m(i)=0.5*(h(i,1)+h(i+1,1))*JLength*dx*oilDen/1000.0, mSum=mSum+m(i),
0082  end;
0083  // Half m(1) is pulled in each slice advance, dx
0084  mInTotPerRev=circInts*0.5*m(1);   // kg
0085  mInTotPerSec=mInTotPerRev*shaftRPM/60.0;   // kg
0086  volInTotPerSec=1000.0*mInTotPerSec/oilDen   //litres
0087  format(15);
0088  disp(mInTotPerRev);
0089  // disp(mInTotPerSec);
0090  // disp(volInTotPerSec);
0091  // Calculate force required to shear oil in each longitudinal slice and work done
0092  // Also calculate temp rise caused by shear energy
0093  startTemp=60.0;
0094  pluscon=53; // factor for equation
0095  temp(1)=startTemp; //in degrees C
0096  TC=0;  //1:1:circInts;
0097  TC(1)=startTemp;
0098  i=1;
0099  // mu0=14*0.0141;
0100  // b=550; // For SAE grade 30 oil
0101  // mU=1:1:circInts;
0102  // mU(1)=(mu0*exp(b/(TC(i)+pluscon)))/1000.0; // Using modified eqn from: A S Seireg
 and S Dandage, ref 1.
0103  // mU(2)=mU(1);
0104  f(1)=2*mU(1)*vel*dx*JLength/(1000.0*(h(1,1)+h(2,1)));
0105  workDOS(1)=dx*f(1);
0106  dTemp1(1)=workDOS(1)/(m(1)*oilSpecHeat);
0107  tempSum=0.0;
0108  for i=2:floor((circInts-1)/2),f(i)=2*mU(i)*vel*dx*JLength/(1000.0*(h(i,1)+h(i+1,1))),
0109  workDOS(i)=dx*f(i), // Work done one slice per 1 slice advance
```

```
0110  dTemp(i)=workDOS(i)/(m(i)*oilSpecHeat),
0111  tempSum=tempSum+dTemp(i), // Temperature rise for slice i
0112  // Add half temp rise to i+1 slice
0113  TC(i)=TC(i-1)+((dTemp(i-1))/2.0)+dTemp(i),
0114  // After each row recalculate viscosity
0115  mU(i)=(mu0*exp(b/(TC(i)+pluscon)))/1000.0;
0116  end;
0117  // Do loop for half of journal circumference where h is increasing, assume constant
 mass
0118  for i=ceil((circInts-1)/2):circInts-1,
0119  f(i)=2*mU(i)*vel*dx*JLength/(1000.0*(h(i,1)+h(i+1,1))),
0120  workDOS(i)=dx*f(i),
0121  dTemp(i)=workDOS(i)/(m(ceil((circInts-1)/2))*oilSpecHeat),
0122  TC(i)=TC(i-1)+((dTemp(i-1))/2.0)+dTemp(i),
0123  end;
0124  frictionTorque=shearForce*JDiam/2000.0;
0125  fricPower=shearForce*vel; // Oil flow rate, 2 terms
0126  oilQ1=vel*radialClear*JLength*eccen/1000000.0;
0127
  oilQ2=(1.2+11.0*oilHoleDia/JLength)*(h(1:1,1:1))*(h(1:1,1:1))*(h(1:1,1:1))*oilSupPress*oilHoleDia/(12
0128  oilQ=oilQ1+oilQ2;
0129  oilTempRise=fricPower/((oilDen*oilQ)*oilSpecHeat);
0130  printf("Friction force, N, is: %f\n",shearForce);
0131  printf("Friction power, w, is: %f\n",fricPower);
0132  printf("Oil flow, litres/s is: %f\n",oilQ*1000.0);
0133  printf("Oil temperature rise, deg. C, is: %f\n",oilTempRise);
0134  for i=1:circInts,
0135  for j=1:widthInts+1, // zero matrices to store iterations. Start pressure distribution
 calcs
0136  p1(i,j)=0.0;
0137  p0(i,j)=0.0; // Calc coefficients cN, cS, cE, cW, cT - north, south, east, west and cT
0138  end, end;
0139  oilVisc=0.0;
0140  for i=1:circInts, for j=1:widthInts+1,cN(i,j)=dx2/dx2dy2,cS(i,j)=cN(i,j), end,end;
0141  for i=2:circInts-1, mU(i)=(mu0*exp(b/(TC(i)+pluscon)))/1000.0, oilVisc=mU(i),
0142  for j=1:widthInts+1,hi3=h(i,j)*h(i,j)*h(i,j),p25=0.25*dy2/hi3,..
0143  hi3p=h(i+1,j)*h(i+1,j)*h(i+1,j),..
0144  hi3m=h(i-1,j)*h(i-1,j)*h(i-1,j),..
0145  cE(i,j)=(dy2+p25*(hi3p-hi3m))/(dx2dy2),..
0146  cW(i,j)=(dy2+p25*(-hi3p+hi3m))/(dx2dy2),..
0147  end; end;
0148  // Next solve matrix of eqns, k is iteration step no.
0149  // To obtain a solution under relaxation must be used and thousands of iterations are
 normally needed
0150  for k=1:iTerations,
0151  for i=2:circInts-1,
0152  f(i)=2*mU(i)*vel*dx*JLength/(1000.0*(h(i,1)+h(i+1,1))),
0153  workDOS(i)=dx*f(i), // Work done one slice per 1 slice advance
0154  dTemp(i)=workDOS(i)/(m(i)*oilSpecHeat),
0155  tempSum=tempSum+dTemp(i), // Temperature rise for slice i
0156  // Add half temp rise to i+1 slice
0157  TC(i)=TC(i-1)+(dTemp(i-1)/2.0)+dTemp(i),
0158  mU(i)=(mu0*exp(b/(TC(i)+pluscon)))/1000.0;
0159  for j=2:widthInts,
0160  hi3=h(i,j)*h(i,j)*h(i,j),
0161  cT(i,j)=3.0*dx*dy2*vel*mU(i)*(h(i+1,j)-h(i-1,j))/(dx2dy2*hi3),
0162  cT(i,1)=3.0*dx*dy2*vel*mU(i)*(h(i+1,j)-h(i-1,j))/(dx2dy2*hi3),
0163  cT(i,widthInts+1)=3.0*dx*dy2*vel*mU(i)*(h(i+1,j)-h(i-1,j))/(dx2dy2*hi3),
0164  p1(i,j)=(p0(i,j-1)+p0(i,j+1))*(cN(i,j))+p0(i+1,j)*cE(i,j)+p0(i-1,j)*cW(i,j)-cT(i,j),
  ///
0165  //replace p0 with p1 using sor factortimes difference from above
0166  if(p1(i,j))<0.0 p0(i,j)=0.0; end;
```

```
// SoR factor normally needs to be less than 1 - start with 0.9
pDiff=p0(i,j)-p1(i,j),
p0(i,j)=p0(i,j)-sOR*pDiff;
if(p1(i,j))<0.0 p1(i,j)=0.0; end;
end;end;end;
// Calculate element pressures as average of 4 corner node pressures
eSum=0.0,
for i=1:circInts,
for j=1:widthInts,
pE(i,j)=0.0; end, end;
for i=1:circInts-1,
for j=1:widthInts,
pE(i,j)=0.25*(p1(i,j)+p1(i+1,j)+p1(i,j+1)+p1(i+1,j+1));
if(pE(i,j))>0.0 eSum=eSum+pE(i,j)*dxdy;end;end;end;
printf("Sum of radial lubricant forces, (N) is: %f\n", eSum);
// Calculate element forces so load parallel and perpendicular to min. film thickness
 can be calculated
// Use negative sign
loadPerp=0.0; loadParallel=0.0;
for i=1:circInts-1,
for j=1:widthInts,
eLoad=pE(i,j)*dxdy;
loadParallel=loadParallel-eLoad*cos(i*dTheta);
loadPerp=loadPerp+eLoad*sin(i*dTheta);
end; end;
printf("Load parallel to minimum film thickness, (N) is: %f\n", loadParallel);
printf("Load perpendicular to minimum film thickness, (N) is: %f\n", loadPerp);
resLoad=sqrt((loadPerp*loadPerp)+(loadParallel*loadParallel));
printf("Resolved load, (N) is: %f\n", resLoad);
// Calculated angle of attitude is in degrees after the downwards vertical - which is
 the direction of the load
calcAt=180.0*atan(loadPerp/loadParallel)/3.14159;
calcAtRad=atan(loadPerp/loadParallel);
calcAtFloor=floor(calcAt);
printf("Calculated angle of attitude (degrees) is: %f\n", calcAt);
// Calculate max and av pressures
maxPress=0.0; maxPresPosni=0; maxPresPosnj=0; ;
for i=2:circInts-1,
for j=1:widthInts+1,
if(p1(i,j))>maxPress, maxPress=p1(i,j), maxPresPosni=i, maxPresPosnj=j; end;
end;end;
maxPressMPa=maxPress/1000000.0;
printf("Maximum pressure is: (MPa) %f\n", maxPressMPa );
ratPAvPMax=avProjPress/maxPressMPa;
printf("Ratio average pressure/max. pressure (load/projected area*max. pressure) is:
 %f\n", ratPAvPMax);
// Calculate angular position of max pressure past downward vertical
maxPressAng=calcAt-180.0+maxPresPosni*dTheta*180.0/3.14159;
printf("Angle of maximum pressure past downwards vertical %f\n\n", maxPressAng );
printf("For diagnostics, type variable name at prompt \n");
printf("eg: for list of pressures on each element at prompt type pE then RTN \n");
// Prepare plot of pressure (vert axis) round cente line (straight, horiz axis)
cT1=0;
cI1=round(360/circInts); //value in degrees of 1 circular interval
xx=[round(2*cT1/circInts):round(cT1/circInts):round(cT1*(circInts-1)/circInts)];
//xx=[round(2*%pi/circInts):round(%pi/circInts):round((circInts-1)*%pi/circInts)];
// xxxx=[4*%pi/180:2*%pi/180:358*%pi/180];
for i=2:circInts-2, yyA(1,i)=pE(i,maxPresPosnj)/1000000.0; end;
figure(2);
plot2d(xx,yyA);
title('Plot of element pressures in MPa along centre line of journal - radians -
 starting at maximum film thickness','fontsize',3);
```

```
0225  xtitle('For angular position past max. oil film thickness multiply by 360/number of
 circular intervals');
0226  // Polar plot of base circle
0227  // xxbase=[round(2*%pi/circInts):round(%pi/circInts):round((circInts-1)*%pi/
circInts)];
0228  xxbase=[4*%pi/180:2*%pi/180:358*%pi/180];
0229  // xxbase=[4:2:358];
0230  yybase=[3.0004:0.0002:3.0358];
0231  // Polar plot of pressure - first determine centre element width number
0232  // Shaft rotates in clockwise direction
0233  widthIntsMax=0;
0234
  if(widthInts-fix(widthInts/2)*2==0),widthIntsMax=widthInts/2,else widthIntsMax=ceil(widthInts/2),end;
0235  // disp(widthIntsMax)
0236  figure(3);
0237  for i=2:circInts-2, yyB(1,i)=3+pE(i,widthIntsMax)/1000000.0; end;
0238  ii=%pi*(90-calcAt)/180;
0239  for i=2:circInts-2,xxx(i)=ii-2*%pi/circInts,ii=ii-2*%pi/circInts,end;
0240  // for i=2:circInts-2,xxx(i)=ii-2*%pi/180,ii=ii-2*%pi/180,end;
0241  polarplot(xxbase,yybase),polarplot(xxx,yyB),;
0242  title('Polar plot of pressure in MPa on centreline of journal. Zero pressure is on 3.
 Load on shaft is down','fontsize',3);
0243  figure(4);
0244  x=[1:widthInts];
0245  y=[1:circInts];
0246  z=pE(y,x);
0247  contour(y,x,z,12);
0248  title('Contour plot of oil pressures on elements MPa','fontsize',3);
0249  xtitle('For angular position past max. oil film thickness multiply by 360/number of
 circular intervals');
0250  // Plot temperatures round journal
0251  figure(5);
0252  for i=2:circInts-2, yyATemp(i)=TC(i),end;
0253  plot2d(xx,yyATemp);
0254  title('Plot of oil temperature round journal journal - degrees C - starting at maximum
 film thickness','fontsize',3);
0255  // xtitle('For angular position past max. oil film thickness multiply by 360/number of
 circular intervals');
0256  mU(circInts)=mU(circInts-1);
0257  figure(6);
0258  plot2d(y,mU);
0259  title('Plot of oil Viscosity, starting at max. oil film thickness','fontsize',3);
```