Scilab Textbook Companion for Fundamentals Of Modern Manufacturing: Materials, Processes, And Systems by Mikell P. Groover[1]

Created by
Venkata Ajay Kumar G
M.Tech - College Teacher
Mechanical Engineering
JNTU Anantapur / AITS, Rajampet
Cross-Checked by
Scilab TBC Team

April 8, 2020

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Fundamentals Of Modern Manufacturing: Materials, Processes, And Systems

Author: Mikell P. Groover

Publisher: John Wiley & Sons, Inc.   Usa

Edition: 5

Year: 2012

ISBN: 978-1-118-231463

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 1

Introduction and Overview of Manufacturing

Scilab code Exa 1.1 Equipment cost rate

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 1.1 - Equipment cost rate
// Given that
ic = 500000 // Initial cost plus installation in $
n = 7 // Anticipated life in years
w = 50 //number of weeks per year
s = 2 // Number of shift
D = 5 // Number of days in week
h = 8 // Shift hours
Roh = 0.35 // Overhead rate on the equipment in Percentage
H = w*s*D*h // Annual number of hours of Operation in hr/yr
Ceq = (ic/(60*n*H))*(1+Roh)  // Equipment cost rate
printf("\n The Equipment cost rate = $%.3f /min",Ceq
```

)

Scilab code Exa 1.2 Cycle time and Cost per piece

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 1.2 - Cycle time and Cost per piece
// Given that
ic = 500000 // Initial cost plus installation in $
n = 7 // Anticipated life in years
w = 50 //number of weeks per year
s = 2 // Number of shift
D = 5 // Number of days in week
h = 8 // Shift hours
Roh = 0.35 // Overhead rate on the equipment in
    Percentage
H = w*s*D*h // Annual number of hours of Operation
    in hr/yr
Ceq = (ic/(60*n*H))*(1+Roh)  // Equipment cost rate
To = 3.72 // Processing time in min
Th = 1.60 // Part handling time in min
p = 20 // Number of pieces changing
t = 2 // setup time in min
Q = 100 // Batch quantity, number of pieces (pc)
Ts = 2.5 // Machine setup time in hr
Rh = 16.50 // Hourly wage rate is $/hr
Rloh = 0.40 // labor overhead rate in percentage
tool_cost = 4.40 // Tool cost in Dollar
Cm = 2.35 // Starting material cost in dollars
Tt = t/p // Tool handling time in min for 20 tools
Tc = To + Th + Tt  // Cycle time of the unit
    operation in min/pc
```

```
Tp = ((Ts*60)/Q)+Tc //Average production time per
    piece inculding the effect of setup time in min/
    pc
Rp = (60/Tp) // Hourly production rate in pc/hr
Cl = Rh/60 *(1+Rloh) //  labor cost rate, $/min;
Ct = tool_cost/20 // Each tool can be used for 20
    pieces in $/pc
Cpc = Cm + ((Cl+Ceq)*Tp) + Ct // Cost per piece in $
    /pc
printf("\n The Cycle time for the piece = %.2f min/
    pc \n Average production rate = %.2f min/pc \n
    Hourly production rate = %.2f pc/hr \n Cost per
    piece = $%.2f /pc",Tc,Tp,Rp,Cpc)
// The answers vary due to round off error
```

Scilab code Exa 1.3 Scrap rate

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 1.3 - Scrap rate
// Given that
Q = 1000 // Ordered batch parts
q = 0.04 // Scrap rate for the type of part in
    percentage
Qo = Q/(1-q) // Required quantity of parts to be
    delivered
printf('Number of parts required = %.0f \n',Qo)
```

Scilab code Exa 1.4 Cycle time and cost per piece

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 1.4 - Cycle time and cost per piece
// Given that
Cm = 1.75 // Starting Material Cost in $
Ceq = 42 // Equipment Cost rate in $/hour
Tp = 0.97 // Average production time per piece in percentage
Cl = 24 // Labor cost rate in $/hr
s = 0.05 // Scrap rate of parts prodcued in Percentage
T_p = 2.20 // Cycle time in min
Rp = (60*Tp)/T_p // Production rate, including effect of availability in pc/hr
average_Rp = Rp*(1-s) // Because of 5% scrap rate the production rate of acceptable parts
Cpc = (Cm/(1-s))+((Cl+Ceq)/Tp)*((T_p)/(60*(1-s))) // Availability of Scrap rate, the part cost per piece
s_b = 0 // Scrap rate of parts prodcued is Zero Percentage for b question
Tp_b = 1 // Average production time per piece in percentage for b question
Rp_b = (60*Tp_b)/T_p // Production rate, including effect of availability in pc/hr for b question
Cpc_b = (Cm)+((Cl+Ceq)/Tp_b)*((T_p)/(60*(1-s_b))) // Availability of Scrap rate, the part cost per piece for b question
printf('The part cost for (A) = $ %.2f /pc \n The part Cost for (B) = $ %.2f \n',Cpc,Cpc_b)
```

Chapter 3

Mechanical Properties of Materials

Scilab code Exa 3.1 Engineering Stress and Strain

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 3.1 - Engineering Stress and Strain
// Given that
F = 32000 // Force applied in N
Ao = 200 // Original area of cross of test speciment in MPa
Y =  F/Ao // Yield Strength (Y), MPa
L = 50.2 // Length at any point during the elongation in mm
Lo = 50 // Gage length in mm
Lo_max = 57.7 // Maximum length before necking beigns in mm
Le = 0.002 //
Fmax = 65000 // Maximum load in N
Lf = 63.5 // Speciemen length at fracture in m
```

```
// The input consider in the textbook was wrong while solving the problem
e = (L-Lo)/(Lo - Le) // Engineering Strain in mm/min , 0.2% offset
E = Y/e  // Modulus of Elasticity , MPa
// the answer in the textbook was wrong
TS = Fmax/Ao  // Tensile strength in MPa
e_max_load = (Lo_max-Lo)/Lo // Engineering Strain at maximum load in mm/min
EL = (Lf - Lo)/Lo // Elongation in %
// the answer in the textbook was wrong
printf('Yield Strength = %.0f MPa \n Modulus of Elasticity = %.0f MPa \n Tensile Strength = %.0f MPa\n Engineering strain at maximum load = %.3f \n Percentage Elongation = %.2f \n',Y,E,TS,e_max_load,EL)
```

Scilab code Exa 3.2 True Stress and Strain

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 3.2 - True Stress and Strain
// Given that - Some of the data are from problem 3.1
F = 32000 // Force applied in N
Ao = 200 // Original area of cross of test speciment in MPa
s =  F/Ao // Yield Strength (Y), MPa
L = 50.2 // Length at any point during the elongation in mm
Lo = 50 // Gage length in mm
Lo_max = 57.7 // Maximum length before necking
```

```
   beigns in mm
Le = 0.002 //
Fmax = 65000 // Maximum load in N
Lf = 63.5 // Speciemen length at fracture in m
A = (Ao*Lo)/Lo_max // Instantaneous area
sigma = Fmax/A // True Stress in MPa
epilson = log(Lo_max/Lo) // True strain
printf('True Stress = %.0f MPa \n True Strain = %.3f
    \n',sigma,epilson)
```

Scilab code Exa 3.3 Flow curve parameters

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 3.3 - Flow curve parameters
// Given that
// Data given from the example 3.1
F = 32000 // Force applied in N
Ao = 200 // Original area of cross of test speciment
    in MPa
Y =  F/Ao // Yield Strength (Y), MPa
L = 50.2 // Length at any point during the
   elongation in mm
Lo = 50 // Gage length in mm
Lo_max = 57.7 // Maximum length before necking
   beigns in mm
Le = 0.002 //
Fmax = 65000 // Maximum load in N
Lf = 63.5 // Speciemen length at fracture in m
e = log((L/Lo)-Le ) // True strain
// At maximum load
e_2 = 0.143
```

```
sigma = 375
// The corresponding flow equation is 160 = K(0.001998)^n for Example 3.1
// the corresponding flow equation is 375 = K(0.143)^n for Example 3.2
// Solving n
n = (log(sigma)-log(Y))/(log(e_2)-(log(e)))
K = Y/(e)^n //  Substituting back into Example 3.1 equation
printf ('The flow curve equation is sigma = %.1f e^%.4f \n',K,n)
```

Chapter 5

Dimensions Surfaces and their Measurement

Scilab code Exa 5.1 Wear allowance on a fixed gage

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
   the stored variables
clc
// Example 5.1 - Wear allowance on a fixed gage
// Given that
hole_diameter = 20 // hole diameter in mm
tolerance = 0.10
wear = 0.025  // 2.5% of wear allowance
Total_tolerance = tolerance+tolerance // Total
   Tolerance band in mm
wear_allowance = wear*Total_tolerance
acceptable_hole_diameter = hole_diameter - tolerance
    // Minimum acceptable hole diameter in mm
nominal_size = acceptable_hole_diameter +
   wear_allowance // Nominal size of GO gage in mm
// NO GO gage is used to check the maximum hole
   diameter
```

```
nominal_size_NOGO  = hole_diameter + tolerance // Nominal size of NOGO gage in mm
printf('Nominal size of GO gage = %.3f mm \n Nominal size of NOGO gage = %.2f mm \n',nominal_size,nominal_size_NOGO)
// Answer varies due to round off error
```

Scilab code Exa 5.2 Sine bar measurement

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 5.2 - Sine bar measurement
// Given that
L = 200 // Length in mm
H = 40.38 // Gage blocks are stacked to a height in mm
A = asind(H/L) // Angle of interest in degrees
printf('The Angle of interest = %.2f degrees',A)
```

Chapter 10

Fundamentals of Metal Casting

Scilab code Exa 10.1 Heating metal for casting

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 10.1 - Heating metal for casting
// Given that
row = 7.5 // Density in g/cm^3
V = 1000000 // m^3 = 10^6 cm^3
C_s = 0.33 // Specific heat in J/gC
T_m = 800 // Melting point in C
T_o = 25 // Ambient temperature in the foundry C
H_f = 160 // Heat of fusion J/g
C_l = 0.29 // Weight specific heat of the liquid
   metal J/gC
T_p = 100 // pouring temperature in C
H = row*V*((C_s*(T_m - T_o))+H_f+(C_l*(T_p)))
printf('Heat energy = %.0f J',H)
```

Scilab code Exa 10.2 Pouring Calculations

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 10.2 - Pouring Calculations
// Given that
g = 981 // acceleration gravity constant
h = 20 // Mold sprue height in cm
v = sqrt(2*g*h)  // Velocity of the flowing metal at
     the base of the sprue in cm/s
a = 2.5  // cross-sectional area of base in cm^2
V = 1560 // Volume of mold cavity in cm^3
Q = a*v  // Volumetric flow rate in cm^3/s
T_mf = V/Q // Mold filling time in sec
printf('Velocity of the flowing metal at the base of
     the sprue = %.1f cm/s \n Volumetric flow rate =
    %.0f cm^3/s \n Mold filling time = %.1f sec',v,Q,
    T_mf)
```

Scilab code Exa 10.3 Riser design using Chvorinov s rule

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 10.3 - Riser design using Chvorinov's
    rule
// Given that
l = 7.5 // Length in cm
w = 12.5 // width in cm
t = 2.0 // thickness in cm
```

```
n = 2 // Exponent usually taken as 2
T_ts = 1.6 // Total solidification time in min
T_tsd = 2 // Next riser must be designed so that its total solidification time in min
V = l*w*t // Volume in cm^3
A = (2*((l*w)+(l*t)+(w*t)))
C_m = T_ts/((V/A)^n) // Mold constant in min/cm^2
// Since D/H = 1.0, D = H
// V = %pi*D^3/4
// Thus V/A ratio = D/6.
D = sqrt(n/((C_m/6^2)))
H = D // Since H = D
printf('The Dimensions of riser = %.1f cm \n The height of the riser = %.1f cm',D,H )
```

Chapter 11

Metal Casting Processes

Scilab code Exa 11.1 Buoyancy in Sand Casting

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 11.1 - Buoyancy in Sand Casting
// Given that
v = 1875 // Sand core volume in cm^3
row = 1.6 // Density in g/cm^3
W_c = v*row  // Weight of the core
row_lead = 11.3 // From table 11.1
W_lead = row_lead * v
F_b = ((W_lead - W_c)*9.81)/1000 // Buoyancy Force
   in N so divided by 1000
printf('Buoyancy Force = %.1f N',F_b)
```

Scilab code Exa 11.2 Rotation speed in true centrifugal casting

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 11.2 - Rotation speed in true centrifugal
     casting
// Given that
D = 0.25 // Diameter of the mold in m
g = 9.8 // Acceleration of gravity
GF = 65  // G-factor
N = (30/%pi)*sqrt((2*g*GF)/D)
printf('Rotational Speed = %.1f rev/min',N)
```

Chapter 13

Shaping Processes for Plastics

Scilab code Exa 13.1 Extrusion flow rates

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
   the stored variables
clc
// Example 13.1 - Extrusion flow rates
// Given that
D = 75 // Diameter of the extruder barrel in mm
d_c = 6 // channel depth in mm
N = 1 // Screw rotational speed
A = 20 // flight angle
p = 7.0*10^6 // head pressure at the end of the
   barrel in Pa
L = 1.9 // Length of the barrel in m
eta = 100 // Viscosity of the polymer in Pa-s
Q_d = 0.5*(%pi^2)*(((D*10^-3)^2))*(d_c*(10^-3))*sind
   (A)*cosd(A)
Q_b = (p*%pi*(D*10^-3)*((d_c*10^-3)^3)*(sind(A)^2))
   /(12*eta*L)
Q_x = (Q_d - Q_b)
a = 53525*10^-9
```

```
printf('volume flow rate of the plastic in the barrel = %.5e m^3/s',Q_x)
// Answer varies due to round off error
```

Scilab code Exa 13.2 Extruder and Die Characteristics

```
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 13.2 - Extruder and Die Characteristics
// Given that
D = 75 // Diameter of the extruder barrel in mm
L = 1.9 // Length of the barrel in m
N = 1 // Screw rotational speed
d_c = 6 // channel depth in mm
A = 20 // flight angle
eta = 100 // Viscosity of the polymer in Pa-s
D_d = 6.5  // Die opening diameter in mm
L_d = 20  // die opening length in mm
Q_max = 0.5*(%pi^2)*((D*10^-3)^2)*N*(d_c*10^-3)*sind(A)*cosd(A)
P_max = (6*%pi*D*10^-3*N*L*eta*cotd(A))/((d_c*10^-3)^2) // Answers may vary due to round off error
K_s = (%pi*(D_d*(10^-3))^4)/(128*eta*(L_d*(10^-3)))
p = (Q_max - K_s)/(Q_max/P_max) - K_s
//Q_x_p = Q_max - (Q_max/P_max)*p
Q_x = Q_max - ((Q_max/P_max)*p)
printf('Q_max = %f m^3/s \n P_max = %.0f Pa\nShape Factor = %2.2e m^5/Ns \n Q_x = %.3e\n p = %e Pa\n',Q_max,P_max,K_s,Q_x,p)
// the answers may vary due to round off error
// Answer in the textbook is wrong
```

Scilab code Exa 13.3 `Shrinkage in Injection Molding`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 13.3 - Shrinkage in Injection Molding
// Given that
S = 0.025 // Shrinkage from Table 13.1
D_p = 80 // Molded part dimension in mm
D_c = D_p + (D_p*S)+(D_p*S^2)
printf('The dimension of the mold cavity that will
    compensate = %.2f mm \n',D_c)
```

Chapter 18

Bulk Deformation Processes in Metalworking

Scilab code Exa 18.1 Flat Rolling

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 18.1 - Flat Rolling
// Given that
w = 300 //width of strip
t = 25 // Thickness in mm
t_r = 22 // reduced thickness in mm
K = 275 // flow curve
n = 0.15
N = 50 // Roll speed in rev/min
d = t - t_r // Draft attempted in the rolling operation in mm
mew = 0.12 // coefficient of friction between the rolls and the work
r = 250 // radius of the roller
d_max = (mew^2)*r  // Maximum possible draft for the
```

```
given coefficient of friction in mm
L = sqrt(r*(t-t_r)) // Contact length in mm
epsilon = log(t/t_r) //
Y_f = (K*(epsilon^n))/1.15  // Average flow stress in MPa
F = Y_f*w*L // Rolling force in N
T = 0.5*F*L*(10^-3) // Torque required to drive each roll in N-m(10^-3)
P = 2*%pi*N*F*L*(10^-3) // Power obtained in N-m/min
P_watts = P* 0.01667
HP = P_watts/745.7 // One horsepower = 745.7 W
printf('Roll Force = %.0f N \nTorque = %.0f N-m \nHorse Power = %.0f hp',F,T,HP)
// Answer in textbook is wrong
```

Scilab code Exa 18.2 Open die forging

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 18.2 - Open die forging
// Given that
D = 50 // diameter
h = 75 //height
h_a = 62 // Intermediate height
F = 0
epsilon = 0.002
mew = 0.1
K = 350
n = 0.17
A = %pi*((D^2)/4) // Area
V = h*%pi*((D^2)/4) // Workpiece volume
Y_f = K*epsilon^n
```

```
K_f = 1 + ((0.4*mew*D)/h)
F = K_f*Y_f*A // forging force in N
epsilon_a = log(h/h_a)  // At height 62 mm
Y_fa = K*epsilon_a^n // At height 62 mm
A_a = V/h_a // At height 62 mm, assuming constant
    volume and neglecting barreling
D_a = sqrt((4*A_a)/%pi) // Diameter at height 62 mm
K_fa = 1 +  ((0.4*mew*D_a)/h_a)
F_a = K_fa*Y_fa*A_a // forging force in N at 62 mm
h_b = 49 // Intermediate height at 49 mm
epsilon_b = log(h/h_b)  // At height 49 mm
Y_fb = K*epsilon_b^n // At height 49 mm
A_b = V/h_b // At height 49 mm, assuming constant
    volume and neglecting barreling
D_b = sqrt((4*A_b)/%pi) // Diameter at height 49 mm
K_fb = 1 +  ((0.4*mew*D_b)/h_b)
F_b = K_fb*Y_fb*A_b // forging force in N at 49 mm
h_c = 36 // Intermediate height at 36 mm
epsilon_c = log(h/h_c)  // At height 36 mm
Y_fc = K*epsilon_c^n // At height 36 mm
A_c = V/h_c // At height 36 mm, assuming constant
    volume and neglecting barreling
D_c = sqrt((4*A_c)/%pi) // Diameter at height 36 mm
K_fc = 1 +  ((0.4*mew*D_c)/h_c)
F_c = K_fc*Y_fc*A_c // forging force in N at 36 mm
printf(' The Forging force at begin = %.0f N \n The
    Forging force a t intermediate height 62mm = %.0f
     N \nThe Forging force a t intermediate height 49
    mm = %.0f N \n The Forging force a t intermediate
     height  36mm = %.0f N \n',F,F_a,F_b,F_c)
// Answer in textbook is wrong
```

Scilab code Exa 18.3 Extrusion Pressures

```
// OS - Windows 10 (64 bit)
```

```
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 18.3 - Extrusion Pressures
// Given that
L = 75
D_o = 25
a = 0.8
b = 1.5
r_x = 4 // Extrusion ration
K = 415 // Strength Coefficient in MPa
n = 0.18 // Strain hardening exponent
epsilon = log(r_x)
epsilon_x = a + (b*log(r_x))
Y_f = (K*(epsilon^n))/(1+n)
p = Y_f*(epsilon_x + ((2*L)/D_o))
L50 = 50
L25 = 25
L0 = 0
p_50 = Y_f*(epsilon_x + ((2*L50)/D_o))
p_25 = Y_f*(epsilon_x + ((2*L25)/D_o))
p_0 = Y_f*(epsilon_x + ((2*L0)/D_o))
printf('The pressure applied at L=75 is %.0f MPa\
    nThe pressure applied at L=50 is %.0f MPa\nThe
    pressure applied at L=25 is %.0f MPa\nThe
    pressure applied at L=0 is %.0f MPa\n',p,p_50,
    p_25,p_0)
```

Scilab code Exa 18.4 Stress and Force in Wire drawing

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
```

```
clc
// Example 18.4 - Stress and Force in Wire drawing
// Given that
alpha = 15
D_o = 2.5 // Starting diameter in mm
D_f = 2 // Final diameter in mm
K = 205 // Strength coefficient in MPa
n = 0.20 // Strain hardening exponent
mew = 0.07 // Coefficient of friction at the work
    and die interface
D = (D_o+D_f)/2  // Average diameter of work during
    drawing in mm
L_c = (D_o-D_f)/(2*sind(alpha)) // Contact length of
     the work with the draw die
phi = 0.88 + 0.12 *(D/L_c)
A_o = (%pi/4)*D_o^2
A_f = (%pi/4)*D_f^2
epsilon = log(A_o/A_f)
Y_f = (K*epsilon^n)/(1+n) // Average flow stress
sigma_d = Y_f*(1+(mew/tand(alpha)))*phi*epsilon
F = A_f*sigma_d
printf('The Draw stress = %.0f MPa\n The Draw force
    = %.1f N\n',sigma_d,F)
// the answers may vary due to round off error
```

Chapter 19

Sheet Metalworking

Scilab code Exa 19.1 Blanking clearance and force

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 19.1 - Blanking clearance and force
// Given that
b = 3.2
D_b = 150 // Die opening diameter in mm
A_c = 0.075 // From Table 19.1 the clearance
   allowance for half-hard cold rolled steel
F_s = 310 // Shear strength
c = A_c*b
d_p = D_b - 2*c // Punch Diameter in mm
L = %pi*D_b // Length of the cut edge
F = F_s*L*b
printf('The appropriate punch and die diameter = %.2
   f & %d \n The Blanking force = %f N',d_p,D_b,F)
```

Scilab code Exa 19.2 Sheet metal Bending

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 19.2 - Sheet-metal Bending
// Given that
TS = 450 // Tensile Strength in MPa
D = 25 // Die opening diameter in mm
alpha = 60 // Bend angle
alpha_d = 120 // Included angle
K_bf = 1.33
w = 44.5 // width
R = 4.75 // Bend radius in mm
K_ba = 0.33
t = 3.2 // Stock thickness in mm
A_b = (2*%pi*(alpha/360))*(R+(K_ba*t))
blank_size = 38 + A_b + 25 // 35 and 25 are shown in
     figure 19.15
F = (K_bf*TS*w*(t^2))/D
printf('The Length of the blank = %.2f mm \n Force =
    %.0f N',blank_size,F)
```

Scilab code Exa 19.3 Cup Drawing

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 19.3 - Cup Drawing
// Given that
D_b = 138 // Starting blank size in mm
```

```
D_p = 75 // Cup inside diameter in mm
t = 2.4 // Stock thickness in mm
DR = D_b/D_p // Drawing ratio
r = (D_b-D_p)/D_b // reduction
check = (t/D_b)*100
printf('The drawing ratio = %.2f \n',DR)
if DR <= 2 then
    printf('The Drawing operation is Feasible')
else
    printf('The Drawing operation is Not-Feasible')
end
// Answer varies due to round off error
```

Scilab code Exa 19.4 Forces in Deep drawing

```
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 19.4 - Forces in Deep drawing
// Given that
D_p = 75
t = 2.4 // Original blank thickness in mm
TS = 300 // Tensile strength in MPa
D_b = 138 // Blank diameter in mm
Y = 175 // Yield strength in MPa
R_d = 6 // Die corner radius
F = %pi*D_p*t*TS*((D_b/D_p)-0.7) // Maximum drawing
   force in N
F_h = 0.015*Y*%pi*((D_b^2)-(D_p +(2.2*t)+(2*R_d))^2)
printf('Maximum drawing force = %.0f N \nHolding
   force = %.0f N',F,F_h)
```

Chapter 20

Theory of Metal Machining

Scilab code Exa 20.1 `Orthogonal Cutting`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
   the stored variables
clc
// Example 20.1 - Orthogonal Cutting
// Given that
alpha = 10 // Rake angle of tool
t_o = 0.50 // Chip thickness before cut in mm
t_c = 1.125 // Chip thickness after cut in mm
r = t_o/t_c // Chip thickness ratio
phi = atand(((r*cosd(alpha))/(1-(r*sind(alpha)))))
r = tand(phi - alpha) + cotd(phi)
printf('The Shear plane angle = %.1f degrees \n The
   Shear strain = %.3f',phi,r)
// The answers may vary due to round off error
```

Scilab code Exa 20.2 `Shear stress in Machining`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 20.2 - Shear stress in Machining
// Given that
alpha = 10 // Rake angle of tool
t_o = 0.50 // Chip thickness before cut in mm
t_c = 1.125 // Chip thickness after cut in mm
r = t_o/t_c // Chip thickness ratio
F_c = 1559 // Cutting force in N
F_t = 1271 // Thrust force in N
w = 3  // Width of orthogonal cutting
phi = atand(((r*cosd(alpha))/(1-(r*sind(alpha)))))
F_s = (F_c*cosd(phi))-(F_t*sind(phi))
A_s = (t_o*w)/(sind(phi)) // Shear plane area
tau = F_s/A_s
printf('The shear strength of the work material = %
    .0f MPa',tau)
```

Scilab code Exa 20.3 Estimating Friction Angle

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 20.3 - Estimating Friction Angle
// Given that
alpha = 10 // Rake angle of tool
t_o = 0.50 // Chip thickness before cut in mm
t_c = 1.125 // Chip thickness after cut in mm
r = t_o/t_c // Chip thickness ratio
phi = atand(((r*cosd(alpha))/(1-(r*sind(alpha)))))
```

```
beta = (2*(45))+alpha - (2*phi)
mu = tand(beta)
printf('The Friction angle = %.1f degrees \n The
    Coefficient of friction = %.2f \n',beta,mu)
// The answers vary due to round off error
```

Scilab code Exa 20.4 Power Relations in machining

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 20.4 - Power Relations in machining
// Given that
t_o = 0.50  // Data from the previous example 20.1
w = 3 // Data from the previous example 20.2
F_c = 1559 // Data from the previous example 20.2
v = 100 // Cutting speed
// In the textbook it was 1557.
P_c = (F_c*v)
U = P_c/(v*10^3*t_o*w)
printf('The cutting power = %.0f J/min \nThe
    specific energy = %.3f N/mm^3 \n',P_c,U)
```

Scilab code Exa 20.5 Cutting Temperature

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 20.5 - Cutting Temperature
```

```
// Given that
row = 3
C = 1000
t_o = 0.50  // Data from the previous example 20.1
w = 3 // Data from the previous example 20.2
F_c = 1559 // Data from the previous example 20.2
v = 100 // Cutting speed
// In the textbook it was 1557.
K = 50
P_c = (F_c*v)
U = P_c/(v*10^3*t_o*w)
v_c = (v*1000)/60
delta_T = ((0.4*U)/(row*10^-6*C))*(((v_c*t_o)/K)
   ^0.333)
printf('The mean temperature = %.0f C',delta_T)
// The answers vary due to round off error
```

Chapter 21

Machining Operations and Machine Tools

Scilab code Exa 21.1 Machining Time in Turning

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
   the stored variables
clc
// Example 21.1 - Machining Time in Turning
// Given that
v = 2000 // Cutting speed
D_o = 120 // Work Diameter
L = 450  // Length of the workpiece in mm
f = 0.25 // feed
d = 2.2  // Depth of cut in mm
T_m = (%pi*D_o*L)/(f*v) // Cutting time in min
R_mr = v*f*d  // Material removal rate in mm^3/s
printf('The Cutting time = %.1f s\nMaterial Removal
   Rate = %.0f mm^3/s',T_m,R_mr)
```

Scilab code Exa 21.2 Machining Time in Drilling

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 21.2 - Machining Time in Drilling
// Given that
v = 0.5 // Cutting speed
D = 20 // Diameter of twist drill
f = 0.22 // feed
theta = 118 // Point angle
t = 15 // thickness of plate
N = (v*1000)/(%pi*D)
f_r = N*f
A = v*D*tand(90 - theta/2)
T_m = (t+A)/f_r
R_mr = (%pi*(D^2)*f_r)/4
printf('The machining time = %.0f s\n Material
   removal rate = %.1f mm^3/s',T_m,R_mr)
// Answer varies due to round off error
```

Scilab code Exa 21.3 Machining Time in Peripheral Milling

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 21.3 - Machining Time in Peripheral
   Milling
// Given that
v = 0.50 // Cutting speed
D = 65 // Diameter of milling cutter
```

```
n_t = 4 // teeth on milling cutter
f = 0.24 // Feed per tooth
t_i = 56 // Thickness intial
t_f = 50 // Thickness final
w = 60 // width
L = 320
d = t_i-t_f // Depth of cut
N = (v*1000)/(%pi*D)
f_r = N*n_t*f
A = sqrt(d*(D-d)) // Approach distance
T_m = (L + A) / f_r
R_mr = w*d*f_r
printf('The machining time = %.1f s\n Material
    removal rate = %.0f mm^3/s',T_m,R_mr)
// The answers vary due to round off error
```

Chapter 22

Cutting Tool Technology

Scilab code Exa 22.1 `Taylor Tool life equation`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 22.1 - Taylor Tool life equation
// Given that
v1 = 160
T1 = 5
v2 = 100
T2 = 41
// 160(5)^n = 100(41)^n
// Taking natural logarithms of each term
n = (log(v1)-log(v2))/(log(T2)-log(T1))
C = v1*(T1)^n
printf('The values of n and C are = %.3f & %.0f',n,C
   )
```

Chapter 23

Economic and Product Design Considerations in Machining

Scilab code Exa 23.1 Machinability

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 23.1 - Machinability
// Given that
n1 = 0.28
n2 = 0.27
c1 = 350
c2 = 440
T = 60
V_60 = (c1/T^n1)
V1_60 =  (c2/T^n2)
MR = V1_60/V_60
printf('The Machinability rating = %.2f',MR)
```

Scilab code Exa 23.2 `Surface Roughness`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 23.2 - Surface Roughness
// Given that
r = 1.2 // Nose radius in mm
v = 100 // Cutting speed
f = 0.25 // Feed
R_i = ((f^2)/(32*r))*1000 // Ideal surface roughness
    , 1000 for micron meters
R_a = 1.25 * R_i // Figure 23.2, the ratio of actual
     to ideal roughness for ductivle material at 100
    m/min is approximately 1.25
printf('The Actual Surface roughness = %.f um \n',
    R_a)
```

Scilab code Exa 23.3 `Determining cutting speeds in machining economics`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 23.3 - Determining cutting speeds in
    machining economics
// Given that
n= 0.125 // Taylor's tool life exponent
C = 70  // From Table 23.2
T_t = 2 // Tool change time in min
C_o = 30/60 // Converting $30/hr to $0.50/min
C_t = 3 // Tooling cost
```

```
v_max = C/((((1/n)-1)*T_t)^n)
v_min = C*((n/(1-n))*(C_o/((C_o*T_t)+C_t)))^n
printf('Cutting speed for maximum production rate =
    %.0f m/min\nCutting Speed for minimum cost = %.0f
    m/min',v_max,v_min)
```

Scilab code Exa 23.4 Production rate and cost in machining economics

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 23.4 - Production rate and cost in
    machining economics
// Given that
n= 0.125  // Taylor's tool life exponent
C = 70  // From Table 23.2
T_t = 2 // Tool change time in min
C_o = 30/60 // Converting $30/hr to $0.50/min
C_t = 3 // Tooling cost
D = 0.1 // Diameter
L = 0.5 // Work part length
f = 0.25  // feed in mm/rev
T_h = 5 // Handling time per piece
v_max = C/((((1/n)-1)*T_t)^n) // Cutting speed for
    maximum production in m/min
v_min = C*((n/(1-n))*(C_o/((C_o*T_t)+C_t)))^n  //
    Cutting speed for minimum production cost per
    piece in m/min
T_m = (%pi*D*L)/(v_max*f*10^-3) // Machining Time in
    minutes
T = (C/v_max)^8
n_p = T/T_m // Number of pieces per tool
T_c = T_h + T_m +(T_t/n_p) // Average production
```

```
cycle time for the operation
R_p = 60/T_c // Hourly production rate
C_c = (C_o*T_h)+(C_o*T_m)+((C_o*T_t)/n_p)+(C_t/n_p)
T_m_min = (%pi*D*L)/(v_min*f*10^-3)
T_min = (C/v_min)^8   // Tool life in minutes
n_p_min = T_min/T_m_min // Number of pieces per tool
T_c_min = T_h + T_m_min +(T_t/n_p) // Average production cycle time for the operation
R_p_min = 60/T_c_min // Hourly production rate
C_c_min = (C_o*T_h)+(C_o*T_m_min)+((C_o*T_t)/n_p_min)+(C_t/n_p_min)

printf('Hourly Production Rate = %.1f pc/hr \n Average cost per piece = %.2f /pc \n Hourly Production Rate for min cutting speed = %.1f pc/hr \n Average cost per piece for minimum cutting speed= %.0f /pc',R_p,C_c,R_p_min,C_c_min)
// Answers may vary due to round of error
```

Chapter 25

Nontraditional Machining and Thermal Cutting Processes

Scilab code Exa 25.1 Electrochemical Machining

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
   the stored variables
clc
// Example 25.1 - Electrochemical Machining
// Given that
C = 3.44*10^-2 // Specific removal rate from Table
   25.1
l = 10 // length of rectangular hole
b = 30 // breadth of rectangular hole
A = l*b // Area of electrode
t = 12 // thickness of plate
I = 1200
eta = 0.95 // Expected efficiency
f_r = (C*I)/A // Feed rate at current lvel of 1200
   Amps
f_r95 = f_r*eta // Actual Feed rate at 95%
   efficiency
```

```
T_m = (t/f_r95)/60
printf('Feed rate at 95 percentage efficiency = %.4f
    mm/s \n Time to machine through the 12-mm plate
    = %.2f min \n',f_r95,T_m)
```

Scilab code Exa 25.2 Electric Discharge Machining

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 25.2 - Electric Discharge Machining
// Given that
K = 664 // Constant of Proportionality
I = 25 // Discharge current in amps
T_m = 1083 // Melting temperature of copper from
    Table 4.1
R_mr = (K*I)/((T_m)^1.23) // Metal removal rate
printf('The Metal removal rate = %.2f mm^3/s',R_mr)
```

Chapter 27

Surface Processing Operations

Scilab code Exa 27.1 Electroplating

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 27.1 - Electroplating
// Given that
A = 125*100 // Surface area of steel part in mm^2
C = 3.42*10^-2 // Plating constant from Table 27.1
E = 0.95 // Cathode efficiency for nickel from Table 27.1
I = 12 // Current
t = 15 // time during which current is applied
V = E*C*I*t*60
d = V/A
printf('The average plating thickness = %.3f mm\n',d)
```

Chapter 28

Fundamentals of Welding

Scilab code Exa 28.1 Power density in Welding

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 28.1 - Power density in Welding
// Given that
d_i = 5 // diameter of inner circle in mm
d_c = 12 // Concentric circle diameter in mm
Q = 3000 // Heat source transfers
n = 0.70 // 70 percentage distribution follows
nc = 0.90 // 90 percentage distribution for
    concentric cirlce
A = (%pi*d_i^2)/4 // Area of inner circle
P = Q*n // Power inside the area in W
PD_inner = P/A
A_concentric = (%pi*(d_c^2 - d_i^2 ))/4 // Area of
    the ring outside the inner circle
P_concentric = Q*nc - P
PD_concentric = P_concentric/A_concentric
printf('Power density for 5mm diameter = %.0f W/mm^2
```

```
\nPower density for 12mm diameter = %.1f W/mm^2
\n',PD_inner,PD_concentric)
// Observation: The power density seems high enough
   for melting in the inner circle, but probably not
    sufficient in the ring that lies outside this
   inner circle
```

Scilab code Exa 28.2 Welding Travel Speed

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 28.2 - Welding Travel Speed
// Given that
T_m = 1760 // Melting point temperature from Table
   28.2
K = 3.33*10^-6 // Constant when Kelvin scale is used
f1 = 0.7 // Heat transfer factor
f2 = 0.5 // Melting factor
R_h = 3500 // Rate of input energy generated by the
   welding power source
A_w = 20 // Cross-sectional area in mm^2
U_m = K*T_m^2
v = (f1*f2*R_h)/(U_m*A_w)
printf('The travel speed = %.2f mm/s \n',v)
// Answers may vary due to round of error
```

Chapter 29

Welding Processes

Scilab code Exa 29.1 Power in arc welding

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 29.1 - Power in arc welding
// Given that
I = 300 // Current in Amperes
E = 20 // Voltage in V
f1 = 0.7 // Heat transfer factor from Table 29.1
f2 = 0.5 // Melting factor
Um = 10 // melting energy for the metal in J/mm^3
P =I*E // Power in the arc-welding operation in W
R_HW = f1*f2*I*E // Rate of heat used for welding in
     J/s
R_VW = R_HW/Um // Volume of rate of metal welded in
    mm^3/s
printf('Power in the arc-welding operation = %.0f W
    \n Rate of heat used for welding = %.0f J/s \n
    Volume of rate of metal welded = %.0f mm^3/s \n',
    P,R_HW,R_VW)
```

```
// the answer in the textbook was wrong
```

Scilab code Exa 29.2 Resistance welding

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 29.2 - Resistance welding
// Given that
I = 12000 // Current in Amps
R = 0.0001 // Resistance in ohms
t = 0.2 // time in seconds
d = 6 // electrodes diameter in mm
T = 3 // thickness in mm
Um = 12 // Melting energy of the metal in J/mm^3
H = (I^2)*R*t // Heat generated in the operation
v = T*((%pi*(d^2))/4) // Volume of the weld nugget (
   assumed disc-shaped)
Hw = v*Um // Heat required to melt this volume of
   metal in J
Remaining_heat = H - Hw
printf('The remaining heat = %.0f J \n ',
   Remaining_heat)
```

Scilab code Exa 29.3 Heat generation in oxyacetylene welding

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
```

```
// Example 29.3 - Heat generation in oxy-acetylene welding
// Given that
Um = 0.3 // Unit energy required to melt the metal
f1 = 0.20 // Heat transfer factor
D = 9 // Work surface diameter
R_H = Um*55*10^6
R_Hj = R_H/3600 // Rate of heat generated in Joules/sec
p = 0.75 // 75% heat from the flame
Q = f1*R_Hj // Heat received at the work surface
A = (%pi*(D)^2)/4
PD = (p*Q)/A
printf('The rate of heat generated by the torch = %.0f J/s \nThe rate of heat received at the work = %.0f J/s\n Power density in the circle = %.1f W/mm^2',R_Hj,Q,PD)
```

Chapter 31

Mechanical Assembly

Scilab code Exa 31.1 Threaded Fasteners

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 31.1 - Threaded Fasteners
// Given that
C_t = 0.22 // Torque coefficient whose value
   typically ranges between 0.15 and 0.25
D = 8 // Nominal bolt diameter
F = 275 // Specified preloaded tension force in N
p = 1.25
T  = C_t*D*F // Required Torque
A_s = (%pi/4)*((D - (0.9382*p))^2) // Area of the
   minor diameter
sigma = F/A_s
printf('The required torque = %.0f N-mm\n The stress
    on the bolt = %.2f MPa \n',T,sigma)
```

Scilab code Exa 31.2 Expansion fit

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 31.2 - Expansion fit
// Given that
alpha = 12*10^-6 // Thermal expansion for steel from
    Table 4.1
D_2 = 30 // Inner diameter in mm
D_1 = 30.015 // Shaft diameter in mm
T_1 = 20 // Room Temperature in C
c = 0.03 // Clearance
T_2 = (((D_2-c) - D_1)/(alpha*D_1))+T_1
E = 209*10^3 // Modulus of elasticity in MPa  from
   Table 3.1
D_c = 50 // Outer diameter of the collar
D_p = 30.025 // Pin diameter
i = 0.015
P_f = (E*i*(D_c^2 - D_p^2))/(D_p*D_c^2)
Max_sigma_e = (2*P_f*D_c^2)/(D_c^2 - D_p^2)
printf('The Temperature to which the shaft must be
   cooled for assembly = %.1f C \n The radial
   pressure at room temperature after assembly = %.1
   f MPa \n The maximuum effective stress on the
   collar = %.0f MPa',T_2,P_f,Max_sigma_e)
```

Chapter 32

Rapid Prototyping and Additive Manufacturing

Scilab code Exa 32.1 `Build cycle time in stereolithography`

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 32.1 - Build cycle time in stereolithography
// Given that
b = 40
A1 = b^2 // Cross-sectional area of the base
t = 5
H = 52 // Height of cup
tl = 0.10 // Layer thickness
T_su = 20 // Setup time
A2 = 40^2 - 32^2
v = 950 // Speed
TH = H - t // Total height
n_l = t/tl // Number of layers to build base
n_b = TH/tl  // Number of layers to build the walls
```

```
DE_p = 0.25 //Spot diameter
T_r = 21 // Repositioning and recoating time for
    each layer
//A2 =
T_i = (A1/(v*DE_p))+T_r
T_i_wall = (A2/(v*DE_p)) + T_r
T_c = (T_su*60)+ (n_l*T_i) + (n_b*T_i_wall)
printf('Total build cycle time = %.0f s',T_c)
```

Scilab code Exa 32.2 Cost per piece in additive manufacturing

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
    the stored variables
clc
// Example 32.2 - Cost per piece in additive
    manufacturing
// Given that
w = 50 // Weeks
d = 5 // days
h = 8 // hours per day
y = 4 // years
C_l = 24 // Labor rate
U_l = 0.25 // labor build cycle
T_c = 3.777 // Cycle time
T_pp = 6/60 // 6 min/part post processing time, 60
    is divided for minutes
h1 = 52
t = 5
H = h1-t // height of wall
b = 40
A1 = b^2 // Cross-sectional area of the base
A2 = b^2 - 32^2
C = 100000 // Cost of Stereolithography machine
```

```
NH = w*d*h // Numbers of hours of operation per year
MC = 120
C_eq = C/(y*NH) // Hourly equipment cost
V1 = d*A1
V2 = H*A2
V = V1+V2
C_m = (MC*10^-6)*(V)
C_pc = C_m + (((C_l*U_l)+C_eq)*T_c)+(C_l*T_pp)
printf('Cost per piece = $ %.2f /pc \n',C_pc)
```

Chapter 33

Processing of Integrated Circuits

Scilab code Exa 33.1 Number of Chips on Water

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 33.1 - Number of Chips on Water
// Given that
D_w = 190 // Diameter of the processable area of the
    wafer
L_c = 18 // Side dimension of the chip
n_c = 0.34*(D_w/L_c)^2.25
printf('IC chips = %.0f chips',n_c)
```

Scilab code Exa 33.2 Rent s Rule

```
// OS - Windows 10 (64 bit)
```

```
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 33.2 - Rent's Rule
// Given that
A = 17^2 // Processable area of the chip
n = 500
C = 0.89 // rents rule parameters
m = 0.45
n_ic = A*n
n_io = C*n_ic^m
C1 = 6.9 // rents rule parameters for 1
m1 = 0.12
n_io1 = C1*n_ic^m1
printf('Number of Circuits = %.0f \n Rents Rule with C=0.89 and m = 0.45 = %.0f input/output terminals \n Rents Rule with C=6.9 and m = 0.12 = %.0f input/output terminals \n',n_ic,n_io,n_io1)
```

Scilab code Exa 33.3 Yield in Water processing

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 33.3 - Yield in Water processing
// Given that
D_w = 190 // Diameter of the processable area of the wafer
L_c = 10 // Side dimension of the chip
D = 0.002
n_c = 0.34*(D_w/L_c)^2.25
A = (%pi*D_w^2)/4
```

```
Y_m = 1/(1+((A/100)*D)) // Converting A in to Cm^2
number_good_chips = Y_m*n_c
printf('Number of Good Chips = %.2f chips',
   number_good_chips)
```

Chapter 37

Automation Technologies for Manufacturing systems

Scilab code Exa 37.1 Open loop Positioning

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
    the stored variables
clc
// Example 37.1 - Open-loop Positioning
// Given that
x = 75
p = 5 // Leadscrew pitch
A_ls = (360*x)/p  // Lead screw rotation angle
r_g = 4
n_s = 48
V_t = 400
n_p = (r_g*n_s*A_ls)/360
N_ls = V_t/p
N_m = r_g*N_ls
f_p = (N_m*n_s)/60
printf('pulses required to move the table the
    specified distance = %.0f pulses\n Motor Speed =
```

```
%.0f rev/min \n Pulse frequency required = %.0f Hz\n',n_p,N_m,f_p)
```

Scilab code Exa 37.2 NC Closed loop Positioning

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 37.2 - NC Closed-loop Positioning
// Given that
x = 75 // Table distance
n_s = 100 // Pulses generated by Optical encoder
p = 5 // Pitch
f_r = 400 // feed rate
r_g = 4 // Gear ratio
n_p = (x*n_s)/p
f_p = (f_r*n_s)/(60*p)
N_ls = f_r/p
N = r_g*N_ls
printf('Pulses received = %.0f pulses\nPulse rate = %.2f Hz\nMotor speed = %.f rev/min',n_p,f_p,N)
```

Scilab code Exa 37.3 Control resolution accuracy and repeatability

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access the stored variables
clc
// Example 37.3 - Control resolution accuracy and repeatability
```

```
// Given that
p = 5 // Pitch
r_g = 4 // Gear ratio
n_s = 48
L = 550
B = 16
sigma = 0.005
CR_1 = p/(n_s*r_g)
CR_2 = L/(2^B - 1)
CR = max(CR_1,CR_2)
accuracy = 0.5*CR + 3*sigma
repeatability = 3*sigma
printf('Control resolution = %.4f mm\nAccuracy = %.4f mm\nRepeatability = %.3f mm\n',CR,accuracy,repeatability)
```

Chapter 38

Integrated Manufacturing Systems

Scilab code Exa 38.1 Manual Assembly Line

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 38.1  - Manual Assembly Line
// Given that
D_a = 90000 // Annual Demand
S_w = 5 // Number of Shifts/wk
H_sh = 8 // Hours/shift
T_r = 0.150 // In question given repositioning time
   as 9 sec, converted to min
R_p = D_a/(50*S_w*H_sh)
E = 0.95
T_wc = 55 // Work content time
E_b = 0.93 // Balancing Efficiency
T_c = (60*E)/R_p
T_s = T_c - T_r
w = T_wc/(T_s*E_b)
```

```
w_min = T_wc/T_c //Minimum possible number of
    workers
printf('Hourly Production rate to meet demand = %.0f
     units/hr \n Number of workers and Workstations
    required = %.0f Workers and %.0f Workstations \
    nIdeal minimum value = %.2f Workers\n',R_p,w,w,
    w_min)
```

Scilab code Exa 38.2 Automated Transfer line

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 38.2  - Automated Transfer line
// Given that
p = 0.01 // Probability of station failure
n = 20  // Number of stations
T_c = 1
T_b = 10 // Breakdown time
F = p*n
T_p= T_c + F*T_b
R_p = 60/T_p
R_c = 60/T_c
E = (T_c/T_p)*100
printf('Average Production rate = %.0f pc/hr\
    nEfficiency = %.1f\n',R_p,E)
```

Scilab code Exa 38.3 Product Proliferation

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
```

```
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 38.3 - Product Proliferation
// Given that
truck_models = 100
wheel_base = 7
basic_engine = 42
axles = 43
transmissions = 62
rear_axles = 162
annual_production = 130000 // Trucks
Possible_combinations = truck_models*wheel_base*
    basic_engine*axles*transmissions*rear_axles
company_produce_trucks = Possible_combinations/
    annual_production
printf('%d years without ever producing the same
    truck twice',company_produce_trucks)
// Answers may vary due to round off error
```

Chapter 39

Process Planning and Production Control

Scilab code Exa 39.1 Make or buy cost comparison

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
    the stored variables
clc
// Example 39.1  - Make or buy cost comparison
// Given that
UC = 2.25 // Unit material cost in $ per unit
DL = 2 // Direct labor cost in $ per unit
LO = 3 // Labor overhead at 150% cost in $ per unit
EO = 1.75 // Equipment fixed cost in $ per unit
VC = 8 // Component vendor cost in $ per unit
TC = 9 // Home factory cost in $ per unit
CC = VC+EO+LO // Company cost
printf('The Cost to the company = %.2f in $',CC)
// If the equipment can be used to produce other
    components for which the internal prices are less
     than the corresponding external quotes, then a
    buy decision makes good economic sense.
```

Chapter 40

Quality Control and Inspection

Scilab code Exa 40.1 X and R charts

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
   the stored variables
clc
// Example 40.1  - X and R charts
// Given that
m = 8
n = 4
A2 = 0.729 // Data from Table 40.2
D3 = 0 // Data from Table 40.2
D4 = 2.282 // Data from Table 40.2
x = [2.008 1.998 1.993 2.002 2.001 1.995 2.004
   1.999]
R = [0.027 0.011 0.017 0.009 0.014 0.020 0.024
   0.018]
s = [1 2 3 4 5 6 7 8] // Sample size
x_bar = mean(x)
R_bar = mean(R)
LCL = x_bar - (A2*R_bar)
UCL = x_bar + (A2*R_bar)
```

```
LCL = D3*R_bar
UCL = D4*R_bar
plot(s,x)
plot(s,R)
xtitle("Control Chart","Sample number, s","X-chart R
   -chart")
```

Scilab code Exa 40.2 Determining the Sigma level of a process

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear, clc if you want to access
   the stored variables
clc
// Example 40.2 - Determining the Sigma level of a
   process
// Given that
N_u = 9056
N_o = 23
N_d = 479
N_du = 226
DPMO = 1000000*(N_d/(N_u*N_o))
DPM = 1000000*(N_d/N_u)
DUPM = 1000000*(N_du/N_u)
sigma = 3.4 // From table 40.3
printf('The Defects per Million Opportunities(DPMO)
   = %.0d \n The Defects per Million (DPM) = %d \n
   The Defective units per millin (DUPM) = %d \nThe
   corresponding sigma level is about = %.1f',DPMO,
   DPM,DUPM,sigma)
// Answer varies due to round off error
```

Scilab code Exa 40.3 Taguchi Loss Function

```
// OS - Windows 10 (64 bit)
// Scilab version - 6.0.2
clear; // Remove clear , clc if you want to access
    the stored variables
clc
// Example 40.3  - Taguchi Loss Function
// Given that
tolerance = 0.04 // (x-N) is the tolerance
M_c = 80 // Manufacturer cost in $
Product_prob = 0.75 // Product probablity
replace_prob = 1 - Product_prob
E_L_x = Product_prob*M_c + replace_prob*0 //
    Excepeted cost of replacement and shipping
k = E_L_x/(tolerance)^2 // Constant of
    proportionality
tolerance1 = 0.01
L_x = k*(tolerance1)^2
printf('Constant of proportionality = %.0f in $\n
   The Significant reduction = %.2f ',k,L_x)
```