In the design of Bolted Connection, calculating the Strength of a Bolt Group using I.C. Method (Instantaneous Center of Rotation) is a nightmare for most engineers. The process involves the determination of the value of ‘C’ that corresponds to the effective number of bolts in a Bolt Group subjected to Eccentric Load. Having no closed-form formula, the value of C is very infuriating to calculate; it requires the use of tables (Tables 7-7 ~ 7-14 in AISC 13th Edition) provided for by AISC Steel Construction Manual. The use of these tables to determine the value of C offers both ease and difficulty. The process becomes difficult when search values do not concur with the values on the tables – requiring designers to resort to ‘inaccurate’ linear interpolation.

Over a long period of time, in almost daily basis, I have struggled using this apparently crude method in my design work. In my desire to make my job much simpler and easier, I wrote Visual Basic Code (VBA) and finally solved the problem on finding the value of C without the use of tables. The program code is based on the discussion on pages 7-6 to 7-8 of AISC Steel Construction Manual 13th Edition. If you are a user of SteelPro’s ‘Connection Design Section’ (as shown), the code that does the calculation of Bolt Group Coefficient is the very same Visual Basic Code below. The code maybe embedded into

<table>
<thead>
<tr>
<th>Bolt Spacing on the Horizontal</th>
<th>Load Eccentricity</th>
<th>Load Angle from the Vertical [degrees]</th>
<th>Bolt Group Coefficient [C]</th>
<th>Bolt Group Coefficient [C]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.500</td>
<td>45.000</td>
<td>9.347</td>
<td>69.229</td>
</tr>
<tr>
<td>2</td>
<td>5.000</td>
<td>35.000</td>
<td>14.628</td>
<td>116.514</td>
</tr>
<tr>
<td>3</td>
<td>2.500</td>
<td>45.000</td>
<td>9.347</td>
<td>69.229</td>
</tr>
<tr>
<td>4</td>
<td>3.000</td>
<td>0.000</td>
<td>40.000</td>
<td>555.154</td>
</tr>
<tr>
<td>6</td>
<td>30.000</td>
<td>45.000</td>
<td>12.956</td>
<td>314.972</td>
</tr>
<tr>
<td>7</td>
<td>12.000</td>
<td>15.000</td>
<td>21.818</td>
<td>314.645</td>
</tr>
<tr>
<td>8</td>
<td>3.000</td>
<td>45.000</td>
<td>9.347</td>
<td>69.229</td>
</tr>
<tr>
<td>9</td>
<td>1.000</td>
<td>45.000</td>
<td>9.347</td>
<td>69.229</td>
</tr>
<tr>
<td>10</td>
<td>1.000</td>
<td>45.000</td>
<td>9.347</td>
<td>69.229</td>
</tr>
</tbody>
</table>
your Excel Workbook and recognized by Excel as user function -- behaving like any
other excel built-in mathematical functions that calculate values in response to user’s
input. With the code implemented to your spreadsheet, all you have to do is to supply the
necessary function arguments into assigned cells in your spreadsheet. Finally, you get rid
of the cumbersome use of Bolt Coefficient Tables.

For complete instructions on how to embed the VBA Code into your Workbook, please
visit my very young blog:

http://engineersviewpoint.blogspot.com/2010/01/test.html
http://engineersviewpoint.blogspot.com/2010/01/test.html

VBA Code for Calculating the Value of Effective Bolt Coefficient

Option Explicit

Type BoltInfo
    Dv As Double
    Dh As Double
End Type

'Effective Bolt Coefficient Function BoltCoefficient(Bolt_Row As Integer, Bolt_Column As Integer,
Row_Spacing As Double, Column_Spacing As Double, Eccentricity As Double, Optional Rotation As Double = 0) As Double
    Dim i, k, n As Integer
    Dim mP, vP, Ro As Double
    Dim Mo, Fy As Double
    Dim xi, yi As Double
    Dim Ri, xl As Double
    Dim yl, Rot As Double
    Dim Rn, iRn As Double
    Dim Rv, Rh As Integer
    Dim Sh, Sv As Double
    Dim Ec As Double
    Dim Delta, Rmax As Double
    Dim BoltLoc() As BoltInfo
    Dim Stp As Boolean
    Dim J As Double
    Dim FACTOR As Double
    Rv = Bolt_Row
    Rh = Bolt_Column
    Sv = Row_Spacing
    Sh = Column_Spacing
    ReDim BoltLoc(Rv * Rh - 1)
    On Error Resume Next
Rot = Rotation * 3.14159265358979 / 180
Ec = Eccentricity * Cos(Rot)

If Ec = 0 Then GoTo ForcedExit

n = 0
For i = 0 To Rv - 1
 For k = 0 To Rh - 1
  y1 = (i * Sv) - (Rv - 1) * Sv / 2
  x1 = (k * Sh) - (Rh - 1) * Sh / 2
  With BoltLoc(n)
    .Dv = x1 * Sin(Rot) + y1 * Cos(Rot) '''Rotate Vertical Coordinate
    .Dh = x1 * Cos(Rot) - y1 * Sin(Rot) '''Rotate Horizontal Coordinate
  End With
  n = n + 1
 Next
Next

Rn = 74 * (1 - Exp(-10 * 0.34)) ^ 0.55
Ro = 0: Stp = False
Do While Stp = False
 Rmax = 0
 For i = 0 To Rv * Rh - 1
  xi = BoltLoc(i).Dh + Ro
  yi = BoltLoc(i).Dv
  Rmax = Application.WorksheetFunction.Max(Rmax, Sqr(xi ^ 2 + yi ^ 2))
 Next
 Mo = 0: Fy = 0
 mP = 0: vP = 0
 J = 0
 For i = 0 To Rv * Rh - 1
  xi = BoltLoc(i).Dh + Ro
  yi = BoltLoc(i).Dv
  Ri = Sqr(xi ^ 2 + yi ^ 2)
  Delta = 0.34 * Ri / Rmax
  iRn = 74 * (1 - Exp(-10 * Delta)) ^ 0.55
  Mo = Mo + (iRn / Rn) * Ri
  Mo = Mo + (iRn / Rn) * Abs(xi / Ri) * Sgn(xi)
  J = J + Ri ^ 2
 Next
 mP = Mo / (Abs(Ec) + Ro)
 vP = Fy
 Stp = Abs(mP - vP) <= 0.0001
 FACTOR = J / (Rv * Rh * Mo)
 Ro = Ro + Abs(mP - vP) * FACTOR / 2
 DoEvents
 Loop
 BoltCoefficient = (mP + vP) / 2
 Exit Function
 ForcedExit:
 BoltCoefficient = Rv * Rh
 End Function