```
ArrivalRateFunction - 1

Public Function ArrRateFunc(t As Double)

''' Insert your NS arrival function here
'***************'



'***************'

End Function

Public Function MyInvRateFunc(CurValue As Double)
' IFG, for testing

Dim TypeOfFunc As Integer
Dim InvValue As Double          ' R^-1(y)
Dim EndTime As Double
Dim Pi As Double

' If sine function
Dim sinA, sinB, sinC As Double
' If piecewise-constant
Dim i, j, k As Integer
Dim EqualSpace As Boolean
Dim NumInts As Integer
Dim IntVals() As Double
Dim FuncVals() As Double
Dim IntEndPts() As Double
Dim IntPt As Integer
Dim CurInt As Integer
' If other???
Dim IntBlockSize As Integer

Pi = WorksheetFunction.Pi

' What kind of NS rate function?  1 is sinusoidal, 2 is piecewise constant, 3 is other
'***************'
TypeOfFunc = 2
'***************'

If TypeOfFunc = 1 Then

    ' sinusoidal function: set parameters (for Stationary, set 'sinB = 0')
    '***************'
    sinA = 10
    sinB = 9.4
    sinC = 0.4
    '***************

    ' (IGNORE: function evaluated)
    'CurValue = sinA + sinB * VBA.Sin(sinC * Pi * CurTime)
    ' r(t) = A + B*sin(C*pi*t)
    '       ==> R(t) = A*t + (B/(C*pi))*(1-cos(C*pi*t))
    '       ==> Tough to invert

ElseIf TypeOfFunc = 2 Then

    ' piecewise constant: set # of intervals
    '***************'
    'IntBlockSize = 10
    'NumInts = 720 / IntBlockSize
    EqualSpace = False  ' if False, then set endpoints manually below
    i = 1
    Do Until IsEmpty(Worksheets("Simulation Parameters").Cells(6 + i, 7)) = True Or Worksheets("Simula
tion Parameters").Cells(6 + i, 7) > NSNPSim.Initial_SimTime
        i = i + 1
        Loop
    NumInts = i - 1
    '***************'

    ' (IGNORE: code to create enpoint, value vectors)
    EndTime = NSNPSim.Initial_SimTime
    ReDim IntVals(1 To NumInts) As Double
```

Read this from the worksheet.

```
Arrival Rate Function = 2

    ReDim FuncVals(0 To NumInts) As Double
    ReDim IntEndPts(0 To NumInts) As Double
    IntEndPts(0) = 0
    IntEndPts(NumInts) = EndTime
    FuncVals(0) = 0

    ' (IGNORE: piecewise constant: define end points)
    If EqualSpace = True Then
        ' (IGNORE: Interval points spaced equally)
        For IntPt = 1 To NumInts - 1
            IntEndPts(IntPt) = IntEndPts(IntPt - 1) + EndTime / NumInts
        Next IntPt
    Else

        ' Set interval endpoints manually
        '***************'
        ''' In this example, Initial_SimTime = 15
        For i = 1 To NumInts - 1
            IntEndPts(i) = Worksheets("Simulation Parameters").Cells(7 + i, 7).value
        Next i
        '***************'

    End If

    ' piecewise constant: define interval values (value at end of interval)
    '***************'
    For i = 1 To NumInts
        IntVals(i) = Worksheets("Simulation Parameters").Cells(6 + i, 8).value
    Next i


    '***************'

    ' (IGNORE: Store function value at each endpoint)
    For IntPt = 1 To NumInts
        FuncVals(IntPt) = FuncVals(IntPt - 1) + IntVals(IntPt) * (IntEndPts(IntPt) - IntEndPts(IntPt -
1))
    Next IntPt

    ' (IGNORE: determine which interval, get R^-1(y))
    IntPt = 0
    Do While CurValue > FuncVals(IntPt)
        IntPt = IntPt + 1
        Loop
    CurInt = IntPt
    InvTime = IntEndPts(CurInt - 1) + (CurValue - FuncVals(CurInt - 1)) / IntVals(CurInt)

Else ' other type of user defined function

    ' USER: define CurValue
    ' e.g. CurValue = 3*CurTime^2
    '***************'
    Dim MidPt As Double
    MidPt = NSNPSim.Initial_SimTime / 2
    CurValue = 5 + MidPt ^ 2 - (CurTime - MidPt) ^ 2
    If CurValue < 0 Then
        CurValue = 0
    End If
    '***************'

End If


' (IGNORE: assign current value of rate function)
MyInvRateFunc = InvTime

End Function
Public Function MyCumRateFunc(CurTime As Double)

Dim TypeOfFunc As Integer
Dim CheckValue As Double          ' R^-1(y)
Dim EndTime As Double
Dim Pi As Double
```

This is the cumulative rate function $\Lambda(t)$

This is the inverse cumulative rate function $\Lambda^{-1}(t)$

this function does the actual heavy work.

```
ArrivalRateFunction - 3


' If sine function
Dim sinA, sinB, sinC As Double
' If piecewise-constant
Dim EqualSpace As Boolean
Dim NumInts As Integer
Dim IntVals() As Double
Dim FuncVals() As Double
Dim IntEndPts() As Double
Dim IntPt As Integer
Dim CurInt As Integer
' If other???
Dim IntBlockSize As Integer

Pi = WorksheetFunction.Pi

' What kind of NS rate function?  1 is sinusoidal, 2 is piecewise constant, 3 is other
'***************'
TypeOfFunc = 2
'***************'

If TypeOfFunc = 1 Then

    ' sinusoidal function: set parameters (for Stationary, set 'sinB = 0')
    '***************'
    sinA = 10
    sinB = 9.4
    sinC = 0.4
    '***************'

    ' (IGNORE: function evaluated)
    'CurValue = sinA + sinB * VBA.Sin(sinC * Pi * CurTime)
    ' r(t) = A + B*sin(C*pi*t)
    '       ==> R(t) = A*t + (B/(C*pi))*(1-cos(C*pi*t))
    '       ==> Tough to invert

ElseIf TypeOfFunc = 2 Then

    ' piecewise constant: set # of intervals
    '***************'
    'IntBlockSize = 10
    'NumInts = 720 / IntBlockSize
    EqualSpace = False  ' if False, then set endpoints manually below
    i = 1
    Do Until IsEmpty(Worksheets("Simulation Parameters").Cells(6 + i, 7)) = True Or Worksheets("Simula
tion Parameters").Cells(6 + i, 7) > NSNPSim.Initial_SimTime
        i = i + 1
        Loop
    NumInts = i - 1
    '***************'

    ' (IGNORE: code to create enpoint, value vectors)
    EndTime = NSNPSim.Initial_SimTime
    ReDim IntVals(1 To NumInts) As Double
    ReDim FuncVals(0 To NumInts) As Double
    ReDim IntEndPts(0 To NumInts) As Double
    IntEndPts(0) = 0
    IntEndPts(NumInts) = EndTime
    FuncVals(0) = 0

    ' (IGNORE: piecewise constant: define end points)
    If EqualSpace = True Then
        ' (IGNORE: Interval points spaced equally)
        For IntPt = 1 To NumInts - 1
            IntEndPts(IntPt) = IntEndPts(IntPt - 1) + EndTime / NumInts
        Next IntPt
    Else

        ' Set interval endpoints manually
        '***************'
        ''' In this example, Initial_SimTime = 15
        For i = 1 To NumInts - 1
            IntEndPts(i) = Worksheets("Simulation Parameters").Cells(7 + i, 7).value
```

This just counts the number of intervals

All this just sets the end points of the intervals

```
ArrivalRateFunction - 4

        Next i
        '***************'

    End If

    ' piecewise constant: define interval values (value at end of interval)
    '***************'
    For i = 1 To NumInts
        IntVals(i) = Worksheets("Simulation Parameters").Cells(6 + i, 8).value
    Next i


    '***************'

    ' (IGNORE: Store function value at each endpoint)
    For IntPt = 1 To NumInts
        FuncVals(IntPt) = FuncVals(IntPt - 1) + IntVals(IntPt) * (IntEndPts(IntPt) - IntEndPts(IntPt -
 1))
    Next IntPt

    ' (IGNORE: determine which interval, get R^-1(y))
    IntPt = 0
    Do While CurTime > IntEndPts(IntPt)
        IntPt = IntPt + 1
        Loop
    CurInt = IntPt
    CheckValue = FuncVals(CurInt - 1) + IntVals(CurInt) * (CurTime - IntEndPts(CurInt - 1))

Else ' other type of user defined function

    ' USER: define CurValue
    ' e.g. CurValue = 3*CurTime^2
    '***************'
    Dim MidPt As Double
    MidPt = NSNPSim_Initial_SimTime / 2
    CurValue = 5 + MidPt ^ 2 - (CurTime - MidPt) ^ 2
    If CurValue < 0 Then
        CurValue = 0
    End If
    '***************'

End If

' (IGNORE: assign current value of rate function)
MyCumRateFunc = CheckValue

End Function
Public Function MyRateFunc(CurTime As Double)
' IFG, for testing

Dim TypeOfFunc As Integer
Dim CurValue As Double
Dim EndTime As Double
Dim Pi As Double

' If sine function
Dim sinA, sinB, sinC As Double
' If piecewise-constant
Dim EqualSpace As Boolean
Dim NumInts As Integer
Dim IntVals() As Double
Dim IntEndPts() As Double
Dim IntPt As Integer
' If other???
Dim IntBlockSize As Integer

Pi = WorksheetFunction.Pi

' What kind of NS rate function? 1 is sinusoidal, 2 is piecewise constant, 3 is other
'***************'
TypeOfFunc = 2
'***************'
```

get interval values from Excel

Store the function value at each point.

```
ArrivalRateFunction - 5

If TypeOfFunc = 1 Then

    ' sinusoidal function: set parameters (for Stationary, set 'sinB = 0')
    '***************'
    sinA = 10
    sinB = 9.4
    sinC = 0.4
    '***************'

    ' (IGNORE: function evaluated)
    CurValue = sinA + sinB * VBA.Sin(sinC * Pi * CurTime)

ElseIf TypeOfFunc = 2 Then

    ' piecewise constant: set # of intervals
    '***************'
    IntBlockSize = 10
    NumInts = 720 / IntBlockSize
    EqualSpace = True    ' if False, then set endpoints manually below
    '***************'

    ' (IGNORE: code to create enpoint, value vectors)
    EndTime = NSNPSim.Initial_SimTime
    ReDim IntVals(1 To NumInts) As Double
    ReDim IntEndPts(1 To NumInts + 1) As Double
    IntEndPts(1) = 0
    IntEndPts(NumInts + 1) = EndTime

    ' (IGNORE: piecewise constant: define end points)
    If EqualSpace = True Then
        ' (IGNORE: Interval points spaced equally)
        For IntPt = 2 To NumInts
            IntEndPts(IntPt) = IntEndPts(IntPt - 1) + EndTime / NumInts
        Next IntPt
    Else

        ' Set interval endpoints manually
        '***************'
        ''' In this example, Initial_SimTime = 15
        IntEndPts(2) = 1.2
        IntEndPts(3) = 1.5
        IntEndPts(4) = 3.9
        IntEndPts(5) = 7#
        IntEndPts(6) = 10.2
        IntEndPts(7) = 11.4
        IntEndPts(8) = 13.6
        '***************'

    End If

    ' piecewise constant: define interval values (value at start of interval)
    '***************'

If IntBlockSize = 10 Then

        IntVals(1) = 4.944
        IntVals(2) = 4.167
        IntVals(3) = 4.833
        IntVals(4) = 5.489
        IntVals(5) = 4.833
        IntVals(6) = 6.144
        IntVals(7) = 4.967
        IntVals(8) = 5.511
        IntVals(9) = 5.889
        IntVals(10) = 5.133
        IntVals(11) = 4.933
        IntVals(12) = 6.244
        IntVals(13) = 7.044
        IntVals(14) = 11#
        IntVals(15) = 11.244
        IntVals(16) = 7.811
        IntVals(17) = 5.778
```

Intervals and points

→ Example.

```
ArrivalRateFunction - 6

        IntVals(18) = 4.822
        IntVals(19) = 5.067
        IntVals(20) = 4.622
        IntVals(21) = 4.967
        IntVals(22) = 4.644
        IntVals(23) = 5.644
        IntVals(24) = 5.5
        IntVals(25) = 5.611
        IntVals(26) = 5.233
        IntVals(27) = 4.367
        IntVals(28) = 5.2
        IntVals(29) = 4.833
        IntVals(30) = 5.2
        IntVals(31) = 5.678
        IntVals(32) = 5.878
        IntVals(33) = 5.344
        IntVals(34) = 4.944
        IntVals(35) = 4.511
        IntVals(36) = 4.278
        IntVals(37) = 4.678
        IntVals(38) = 5.511
        IntVals(39) = 4.489
        IntVals(40) = 5.311
        IntVals(41) = 5.544
        IntVals(42) = 5.133
        IntVals(43) = 4.344
        IntVals(44) = 5.211
        IntVals(45) = 4.278
        IntVals(46) = 4.589
        IntVals(47) = 4.644
        IntVals(48) = 4.911
        IntVals(49) = 4.533
        IntVals(50) = 4.511
        IntVals(51) = 4.356
        IntVals(52) = 4.189
        IntVals(53) = 4.956
        IntVals(54) = 5.856
        IntVals(55) = 4.667
        IntVals(56) = 4.156
        IntVals(57) = 4.078
        IntVals(58) = 3.789
        IntVals(59) = 4.489
        IntVals(60) = 4.522
        IntVals(61) = 3.833
        IntVals(62) = 4.456
        IntVals(63) = 2.933
        IntVals(64) = 2.922
        IntVals(65) = 2.844
        IntVals(66) = 2.811
        IntVals(67) = 2.856
        IntVals(68) = 2.933
        IntVals(69) = 3.167
        IntVals(70) = 3.044
        IntVals(71) = 3.611
        IntVals(72) = 3.522

ElseIf IntBlockSize = 15 Then

        IntVals(1) = 4.681
        IntVals(2) = 4.615
        IntVals(3) = 5.363
        IntVals(4) = 5.615
        IntVals(5) = 5.193
        IntVals(6) = 5.719
        IntVals(7) = 5.096
        IntVals(8) = 5.778
        IntVals(9) = 8.2
        IntVals(10) = 11.326
        IntVals(11) = 7.326
        IntVals(12) = 4.948
        IntVals(13) = 4.881
        IntVals(14) = 4.889
        IntVals(15) = 4.926
```

```
ArrivalRateFunction - 7

        IntVals(16) = 5.6
        IntVals(17) = 5.378
        IntVals(18) = 4.763
        IntVals(19) = 5.126
        IntVals(20) = 5.03
        IntVals(21) = 5.8
        IntVals(22) = 5.467
        IntVals(23) = 4.741
        IntVals(24) = 4.415
        IntVals(25) = 5.2
        IntVals(26) = 4.585
        IntVals(27) = 5.304
        IntVals(28) = 5.356
        IntVals(29) = 4.622
        IntVals(30) = 4.6
        IntVals(31) = 4.63
        IntVals(32) = 4.8
        IntVals(33) = 4.437
        IntVals(34) = 4.496
        IntVals(35) = 4.015
        IntVals(36) = 5.985
        IntVals(37) = 4.407
        IntVals(38) = 4.193
        IntVals(39) = 4.081
        IntVals(40) = 4.452
        IntVals(41) = 3.689
        IntVals(42) = 3.793
        IntVals(43) = 2.881
        IntVals(44) = 2.837
        IntVals(45) = 2.867
        IntVals(46) = 3.104
        IntVals(47) = 3.2
        IntVals(48) = 3.585

ElseIf IntBlockSize = 30 Then

        IntVals(1) = 4.648
        IntVals(2) = 5.489
        IntVals(3) = 5.456
        IntVals(4) = 5.437
        IntVals(5) = 9.763
        IntVals(6) = 6.137
        IntVals(7) = 4.885
        IntVals(8) = 5.263
        IntVals(9) = 5.07
        IntVals(10) = 5.078
        IntVals(11) = 5.633
        IntVals(12) = 4.578
        IntVals(13) = 4.893
        IntVals(14) = 5.33
        IntVals(15) = 4.611
        IntVals(16) = 4.715
        IntVals(17) = 4.467
        IntVals(18) = 5#
        IntVals(19) = 4.3
        IntVals(20) = 4.267
        IntVals(21) = 3.741
        IntVals(22) = 2.859
        IntVals(23) = 2.985
        IntVals(24) = 3.393

End If


    'IntVals(1) = 0.3
    'IntVals(2) = 1.4
    'IntVals(3) = 2#
    'IntVals(4) = 0.05
    'IntVals(5) = 1.4
    'IntVals(6) = 0.6
    'IntVals(7) = 0.7
    'IntVals(8) = 1#
```

```
ArrivalRateFunction - 8

    '' crazy long data from Zaine and Berko (1992)
    'IntVals(1) = 0.00024
    'IntVals(2) = 0.00016
    'IntVals(3) = 0.00018
    'IntVals(4) = 0.00016
    'IntVals(5) = 0.00026
    'IntVals(6) = 0.00006
    'IntVals(7) = 0.00012
    'IntVals(8) = 0.00018
    'IntVals(9) = 0.0001
    'IntVals(10) = 0.00006
    'IntVals(11) = 0.00008
    'IntVals(12) = 0.00004
    'IntVals(13) = 0.00004
    'IntVals(14) = 0.00008
    'IntVals(15) = 0#
    'IntVals(16) = 0.00006
    'IntVals(17) = 0.00008
    'IntVals(18) = 0.00008
    'IntVals(19) = 0.00002
    'IntVals(20) = 0.00004
    'IntVals(21) = 0.00002
    'IntVals(22) = 0.00008
    'IntVals(23) = 0.00004
    'IntVals(24) = 0.00006
    'IntVals(25) = 0.00002
    'IntVals(26) = 0#
    'IntVals(27) = 0.00006
    'IntVals(28) = 0.00002
    'IntVals(29) = 0.00002
    'IntVals(30) = 0.00002

    '***************'

    ' (IGNORE: function evaluated)
    IntPt = 1
    Do While CurTime > IntEndPts(IntPt + 1)
        IntPt = IntPt + 1
        Loop
    CurValue = IntVals(IntPt)

Else ' other type of user defined function

    ' USER: define CurValue
    ' e.g. CurValue = 3*CurTime^2
    '***************'
    Dim MidPt As Double
    MidPt = NSNPSim.Initial_SimTime / 2
    CurValue = 5 + MidPt ^ 2 - (CurTime - MidPt) ^ 2
    If CurValue < 0 Then
        CurValue = 0
    End If
    '***************'

End If

' (IGNORE: assign current value of rate function)
MyRateFunc = CurValue

End Function
```