NSNPSim - 1

```
' Simulation NSNP: Inversion Technique, As Arrivals to Single-Server Node
Option Private Module
Option Explicit

' Number of Nodes
Const NumNodes = 3

' Events
Const Event_Arrival = 1                    ' ext arr to Q1
Const Event_SvComp = 2                       ' Service Completion Events at Nodes: node+1
Const Event_End_Of_Simulation = 3

' Attributes
Const CurrArrPhase = 3
Const CurrNode = 4
''' Current phase of service for Nodes: node+4

' View through Queue or Just Arrivals
Private ViewThrough As Boolean
''' Service Props: NSNP/M/1
Private NumServ As Integer
Private ServRate As Double
Private MeanSvTime As Double

''' Random Rate Function: using gamma(n,1)
Private WithDSPP As Boolean
Private GammaN As Integer
Private XVar As Double  ' XVar \sim Gamma(GammaN,1)
Private XVarRec() As Double ' keep track of all XVar values

' Lists
''' Queue list label for Nodes: 2*node-1
''' # of Busy Servers list label for Nodes: 2*node

Const Serve_Queue = 1                      ' queue length at Q1
Const Serve_Check = 2                      ' # of busy servers at Q1

' Random number declarations
Const CompMixProb = 1                        ' Checking which Erlang or Expon to use
Const ExponStream = 2                        ' Any exponential generation uses this stream
Const AcceptReject = 3                       ' The check on whether to accept arrival
Const ServTime = 4                           ' For generating service time of single-server
'Const Stream_ExtArrQ1 = 1
''' Stream label for begin service at Nodes: 2*node
''' Stream label for end service at Nodes: 2*node+1

'   Inputs
Private AFirstRow As Integer
Private AFirstCol As Integer

'' Non-Stat Arrival Process
'Private rateA As Double
'Private rateB As Double
'Private rateC As Double

' Pieces to Match
'Private PercOverMax As Double                ' % over max r(t) to set r*
'Private rStar As Double                      ' rate of majorizing process
'Private cvX As Double                        ' cv of majorizing process
'Private skewX As Double                      ' skewness of majorizing process
'Private MomsToMatch As Double                ' # of moments to match: either 2 (only mean & cv) or 3 (
mean, cv, and skew)
'Private MaxArrFunc As Double                 ' evaluated max of r(t)

' Real Pieces to Match
Private scvX As Double
Private rho1X As Double

''' Diff Ph processes to use
Private MajPhType As Integer                ' 1 is Exp, 2 is h2b, 3 is MECon, 4 is MECO (3-moment)
Private SteadyVector() As Double
Private SteadyCumVector() As Double
' Exponential
```

```
Private ExponMean As Double
' h2b
Private h2bRate As Double
Private h2bAlpha As Double
' MECon
Private MEConOrder As Integer
Private MEConRate As Double
Private MEConAlpha As Double
' MECO
Private MECOOrder As Integer
Private MECORate1 As Double
Private MECORate2 As Double
Private MECOAlpha As Double
' Markov-MECO
Private MMECOOrder As Integer
Private MMECORate1 As Double
Private MMECORate2 As Double
Private MMECOAlpha12 As Double
Private MMECOAlpha21 As Double
Private PrevGenErlang As Integer                ' which Erlang generated previous interarrival time
Private MMECOCInf As Double

' Write out majorizing process
Private PhTypeString As String
Private ParamSet1 As String
Private ParamSet2 As String
Private ParamSet3 As String

Private CanMakeMMECO As Boolean

'Private Amatrix() As Double
'Private Acum() As Double                        ' cdf for rows of Amatrix
'Private ArrPh As Integer
'Private Alpha() As Double
'Private AlphaCum() As Double
'Private Lambda() As Double

'Private LambdaConst() As Double
'Private LambdaAmp() As Double
'Private LambdaPer() As Double

''' Service Processes at each Node
'Private Bmatrix() As Double
'Private Bcum() As Double                        ' cdf for rows of Bmatrix
'Private SvPh() As Integer
'Private Beta() As Double
'Private BetaCum() As Double
'Private Mu() As Double
'Private NumServ() As Integer
''Private NumCap() As Integer

' Perf Measures
Private CurMult As Integer
Private MaxMult As Integer
'Private Node1() As Integer                      ' 2-d for # of custs in node 1 at each time in each re
p
'Private Node2() As Integer                      ' 2-d for # of custs in node 2 at each time in each re
p
Private NodeTrace() As Double                   ' 4-d array of counter, time, # of custs, node occurri
ng at each change
Private SampNodeTraces() As Double
Private TotalNodeTraces() As Double
Public OutputArrivalTimes As Boolean
Private EN1st() As Double                       ' 2-d avg of 1st moment of number of custs at each nod
e at each time step
Private EN2nd() As Double                       ' 2-d avg of 2nd moment of number of custs at each nod
e at each time step
Private VarN() As Double                        ' 2-d avg of var of number of custs at each node at ea
ch time step
'Private ED1st() As Double                       ' 2-d avg of 1st moment of number of deps from each n
ode on each time step
'Private ED2nd() As Double                       ' 2-d avg of 2nd moment of number of deps from each n
ode on each time step
```

```
'Private VarD() As Double                          ' 2-d avg of var of number of deps from each node on
each time step
Private RelErrorNum() As Double                   ' 2-d rel error of number of customer estimates at eac
h node
Private VarMeanRat() As Double                   ' ratio of variance to mean of counts at each time
Private RateFuncValue() As Double                 ' value of arr. rate function at each step
Private DerivApprox() As Double                   ' change in estimated E(N(t)) divided by step size
Private MaxInvTime As Double                      ' last time generated by inversion technique across al
l reps

' new variables
Private Number_Reps As Integer
Private StepSize As Double
Private NumberOfSteps As Integer
Private AdjNumberOfSteps As Integer
Public Initial_SimTime As Double
Public Adj_SimTime As Double
'Private StepCounter As Integer                    ' which interval we are currently in
Private MyPi As Double                            ' so I have the value of PI
Private NodeCount As Integer                     ' incremented every time # of custs at each node change
s
Private NodeLength() As Integer                   ' # of counts for each rep and for each node
'Private LastNumNodes As Integer
Private LastS As Double

Private Simulation_StartClock As Double
Private Simulation_EndClock As Double

Private IntNote As Double

Private NodeSum() As Double
Private NodeSqSum() As Double
'Private NodeDepsSum() As Double
'Private NodeDepsSqSum() As Double
Private BigTable() As Double
'' Summary

Private MyNumber As Integer

Public Sub NSNPSim_Main()

    Dim j As Integer
    Dim Sim_RunTime As Double

    'Initialize the random seeds
    Call InitializeRNSeed

    ' Specify Stationary Majorizing Process
    'Call GetMajPh
    Call NewGetMMECO
    If CanMakeMMECO = False Then
        Exit Sub
    End If

    '/* Initialize the model. */
    Call NSNPSim_BeginModel

    '/* Set maxatr = max(maximum number of attributes per record, 4) */
    maxatr = 8

    CurMult = 1

    For j = 1 To Number_Reps

        '/* Initialize simlib */
        Call init_simlib

        '/* Initialize the model. */
        Call StartRep(j)

        ' Run the simulation until the end event
        Do
```

```
            '/* Determine the next event. */
            Call timing

            '/* Invoke the appropriate event function. */
            Select Case next_event_type
            Case Event_Arrival:
                Call MakeArrival(MajPhType, j)
            Case Event_SvComp
                Call CompleteService(j)
'            Case Event_Q2_ExtArr:
'                Call ExtArr_Q2(j)
'            Case Event_Q1_ServComp:
'                Call SvComp_Q1(j, (transfer(SvPhaseQ1)))
'            Case Event_Q2_ServComp:
'                Call SvComp_Q2(j, (transfer(SvPhaseQ2)))
'            Case Event_Q2_DepQ1:
'                Call ArrQ2_DepQ1(j)
            'Case Event_Q2_DepSystem:
            '    Call DepSystem((transfer(SvPhaseQ2)))
            'Case Event_UpdateNumbers:
            '    Call NodeTotals(j)
            Case Event_End_Of_Simulation:
                Call End_Simulation(j)
            End Select

        Loop Until next_event_type = Event_End_Of_Simulation

    Next j

    'output sim length
    Worksheets("Progress").Cells(20, 2).value = "Run-Time"
    Simulation_EndClock = Now
    Sim_RunTime = Simulation_EndClock - Simulation_StartClock
    Worksheets("Progress").Cells(21, 2).value = Sim_RunTime

    ' Get averages over reps
    Call OutputAllReps
    'Call GetResults

    'Worksheets("Simulation Parameters").Select

End Sub
Private Sub NewGetMMECO()

Dim i, j, k, R, s As Integer
Dim l As Long
Dim CurNode As Integer

Dim ArrString As String
''Dim SvString() As Variant
'Dim MeanSvTime As Double
''Dim SCV As Double
' For MECO
Dim ProbMoms
Dim MinN1, MinN2 As Double
Dim mecoA, mecoB, mecoC, mecoX, mecoY As Double
Dim mecoM1, mecoM2, mecoM3 As Double
Dim mecoRoot1, mecoRoot2 As Double
' To get Max r(t)
Dim CurVal As Double
Dim CurMax As Double
Dim TheTime As Double
Dim TempMax As Double
Dim LargeNumSteps As Long
Dim SmallIntDiv As Integer

' new terms for Markov-MECO
Dim TwoMomsOrder As Integer
Dim TMat() As Double        ' this is D0 \equiv U(A1-I)
Dim TMatInv() As Double
Dim TMatSqInv() As Double
Dim TMatCuInv() As Double
```

definición de Variables

```
NSNPSim - 5

Dim ZetaVec() As Double      ' this is zeta vec
Dim TrueVarMoms As Double    ' this is true var
Dim TrueSecMoms As Double    ' this is true m2
Dim ImpThirdMoms As Double   ' this is implied m3
Dim SkewX As Double          ' this is standardized third moment
Dim Coeffvar As Double       ' this is cv = sqr(scvx)
Dim NeedBiggerOrder As Boolean
Dim tempMMECOCov As Double

CanMakeMMECO = True

' Simulation Parameters
Number_Reps = Worksheets("Simulation Parameters").Range("NSNR.RepNum").value
Initial_SimTime = Worksheets("Simulation Parameters").Range("NSNR.SimEndTime").value
scvX = Worksheets("Simulation Parameters").Range("NSNR.TargSCV").value
rho1X = Worksheets("Simulation Parameters").Range("NSNR.TargRho1").value

ReDim TotalNodeTraces(Initial_SimTime * 100, Number_Reps) As Double

IntNote = 10
'Adj_SimTime = MyInvRateFunc(Initial_SimTime)
StepSize = 0.01
'StepSize = Worksheets("Progress").Range("NSNP.Step").value
MaxMult = 10

If Initial_SimTime / StepSize + 1 > 15000 Then
    StepSize = StepSize * 10
End If
If IntNote > Number_Reps / 2 Then
    IntNote = Int(Number_Reps / 2)
End If
If IntNote < Number_Reps / 200 Then
    IntNote = Int(Number_Reps / 200)
End If
NumberOfSteps = Initial_SimTime / StepSize + 1

MeanSvTime = 1
TrueVarMoms = scvX * MeanSvTime ^ 2
TrueSecMoms = TrueVarMoms + MeanSvTime ^ 2
Coeffvar = VBA.Sqr(scvX)
' Get Implied Third Moment
If scvX = 1 Then
    ' exponential
    TwoMomsOrder = 1
    ImpThirdMoms = 6

ElseIf scvX < 1 Then
    ' MECon

    ' determine K
    k = 1
    Do While 1 / k > scvX
        k = k + 1
        Loop
    ' Parameters
    MEConOrder = k
    MEConAlpha = (1 / (1 + scvX)) * (k * scvX - VBA.Sqr(k * (1 + scvX) - k ^ 2 * scvX))
    MEConRate = (k - MEConAlpha) / MeanSvTime
    ' Steady State Vector
    ReDim SteadyVector(1 To 2 * k - 1) As Double
    ReDim SteadyCumVector(1 To 2 * k - 1) As Double
    For i = 1 To k
        SteadyVector(i) = MEConAlpha / (k - 1 + MEConAlpha)
    Next i
    For i = k + 1 To 2 * k - 1
        SteadyVector(i) = (1 - MEConAlpha) / (k - 1 + MEConAlpha)
    Next i

    TwoMomsOrder = 2 * MEConOrder - 1
    ReDim TMat(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double       ' this is D0 \equiv U(A1-I)
    ReDim TMatInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatSqInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatCuInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
```

user parameter del usuario

info para el rate function

$E(X^3) = \left(\frac{1}{\lambda}\right)^3 3!$, with $\lambda = 1$

→ Mixture of Erlangs of common order

gerhardt and Nelson 2009 pag 640

gerhardt and Nelson 2009 pag 635

```
    ReDim ZetaVec(1 To TwoMomsOrder) As Double     ' this is zeta vec
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            TMat(i, j) = 0
            TMatInv(i, j) = 0
            TMatSqInv(i, j) = 0
            TMatCuInv(i, j) = 0
        Next j
        ZetaVec(i) = 0
    Next i
    ZetaVec(1) = 1 - MEConAlpha
    ZetaVec(MEConOrder + 1) = MEConAlpha
    ' TMatInv directly: relative easy
    For i = 1 To MEConOrder
        For j = i To MEConOrder
            TMatInv(i, j) = -1 / MEConRate
        Next j
    Next i
    For i = MEConOrder + 1 To 2 * MEConOrder - 1
        For j = i To 2 * MEConOrder - 1
            TMatInv(i, j) = -1 / MEConRate
        Next j
    Next i

Else
    ' h2b
    TwoMomsOrder = 2
    ReDim TMat(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double       ' this is D0 \equiv U(A1-I)
    ReDim TMatInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatSqInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatCuInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim ZetaVec(1 To TwoMomsOrder) As Double     ' this is zeta vec
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            TMat(i, j) = 0
            TMatInv(i, j) = 0
            TMatSqInv(i, j) = 0
            TMatCuInv(i, j) = 0
        Next j
        ZetaVec(i) = 0
    Next i

    ' get h2b params
    ReDim SteadyVector(1 To TwoMomsOrder) As Double
    ReDim SteadyCumVector(1 To TwoMomsOrder) As Double
    For i = 1 To TwoMomsOrder
        SteadyVector(i) = 1 / TwoMomsOrder
    Next i
    ' Parameters
    h2bAlpha = 0.5 * (1 + VBA.Sqr(1 - 2 / (scvX + 1)))
    h2bRate = 2 * h2bAlpha / MeanSvTime

    ' build T, zeta
    TMat(1, 1) = -h2bRate
    TMat(2, 2) = -h2bRate * (1 - h2bAlpha) / h2bAlpha
    ZetaVec(1) = h2bAlpha
    ZetaVec(2) = 1 - h2bAlpha
    ' invT for h2b is easy b/c T is diagonal
    For i = 1 To TwoMomsOrder
        TMatInv(i, i) = 1 / TMat(i, i)
    Next i

End If

If scvX > 1 Or scvX < 1 Then
    ' calc implied third moment
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            For k = 1 To TwoMomsOrder
                TMatSqInv(i, j) = TMatSqInv(i, j) + TMatInv(i, k) * TMatInv(k, j)
            Next k
        Next j
    Next i
```

Hyperexponential

gerhardt and nelson 2009
pg 640
$CV^2 > 1$

this should be easy to see

```
NSNPSim - 7

    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            For k = 1 To TwoMomsOrder
                TMatCuInv(i, j) = TMatCuInv(i, j) + TMatInv(i, k) * TMatSqInv(k, j)
            Next k
        Next j
    Next i
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            ImpThirdMoms = ImpThirdMoms - 6 * ZetaVec(i) * TMatCuInv(i, j)
        Next j
    Next i
End If

' find minimum Markov-MECO order
SkewX = (ImpThirdMoms - 3 * MeanSvTime * TrueSecMoms + 2 * MeanSvTime ^ 3) / (TrueVarMoms ^ (3 / 2))

' Specfy Markov-MECO to match
' type
MajPhType = 5
If Coeffvar > 0 And SkewX >= Coeffvar - 1 / Coeffvar Then
    MinN1 = 1 / scvX
    MinN2 = (-SkewX + 1 / (Coeffvar ^ 3) + 1 / Coeffvar + 2 * Coeffvar) / (SkewX - (Coeffvar - 1 / Coe
ffvar))

    ' Check if given order feasible
    If MinN1 > MinN2 Then
        k = Int(MinN1) + 1
    Else
        k = Int(MinN2) + 1
    End If
Else
    ProbMoms = VBA.MsgBox("Bad MECO", vbOKCancel + vbCritical)
    CanMakeMMECO = False
    Exit Sub
End If

NeedBiggerOrder = True
Do While NeedBiggerOrder = True
    mecoM1 = MeanSvTime
    mecoM2 = TrueSecMoms
    mecoM3 = ImpThirdMoms

    mecoX = mecoM1 * mecoM3 - ((k + 2) / (k + 1)) * mecoM2 ^ 2
    mecoY = mecoM2 - ((k + 1) / k) * mecoM1 ^ 2
    mecoA = k * (k + 2) * mecoM1 * mecoY
    mecoB = -(k * mecoX + k * ((k + 2) / (k + 1)) * mecoY ^ 2 + (k + 2) * mecoY * mecoM1 ^ 2)
    mecoC = mecoM1 * mecoX

    mecoRoot1 = (-mecoB + VBA.Sqr(mecoB ^ 2 - 4 * mecoA * mecoC)) / (2 * mecoA)
    mecoRoot2 = (-mecoB - VBA.Sqr(mecoB ^ 2 - 4 * mecoA * mecoC)) / (2 * mecoA)

    MECOOrder = k
    MECORate1 = 1 / mecoRoot1
    MECORate2 = 1 / mecoRoot2
    MECOAlpha = ((mecoM1 / MECOOrder) - mecoRoot2) / (mecoRoot1 - mecoRoot2)

    ' now get Markov-MECO stuff
    tempMMECOCov = TrueVarMoms * rho1X
    MMECOAlpha21 = MECOAlpha - tempMMECOCov / ((1 - MECOAlpha) * (MECOOrder * mecoRoot1 - MECOOrder *
mecoRoot2) ^ 2)
    MMECOAlpha12 = MMECOAlpha21 * (1 - MECOAlpha) / MECOAlpha
    MMECORate1 = MECORate1
    MMECORate2 = MECORate2
    ' check feasibility of M-MECO probs
    If MMECOAlpha21 < 0 Or MMECOAlpha21 > 1 Or MMECOAlpha12 < 0 Or MMECOAlpha12 > 1 Then
        k = k + 1
        If k > 4000 Then
            ProbMoms = VBA.MsgBox("Cannot match target moments, check feasible rho1 for this scv.", vb
Critical)
            CanMakeMMECO = False
            'Stop
            Exit Sub
```

```
NSNPSim - 8
        End If

    Else
        NeedBiggerOrder = False
        MMECOOrder = MECOOrder
    End If

    Loop

  Steady State Vector
ReDim SteadyVector(1 To 2 * MMECOOrder) As Double
ReDim SteadyCumVector(1 To 2 * MMECOOrder) As Double
For i = 1 To MMECOOrder
    SteadyVector(i) = (MECORate2 * MMECOAlpha21) / (MMECOOrder * (MECORate1 * MMECOAlpha12 + MECORate2
 * MMECOAlpha21))
Next i
For i = MMECOOrder + 1 To 2 * MMECOOrder
    SteadyVector(i) = (MECORate1 * MMECOAlpha12) / (MMECOOrder * (MECORate1 * MMECOAlpha12 + MECORate2
 * MMECOAlpha21))
Next i
' from survey paper
MMECOCInf = scvX * (1 + 2 * MECOAlpha * rho1X / MMECOAlpha21)

' Model String
PhTypeString = "Majorizing Process is Markov-MECO(" & Format(MMECOOrder, "##,##0") & ")"
ParamSet1 = "Rates: " & Format(MMECORate1, "##,##0.00") & ", " & Format(MMECORate2, "##,##0.00")
ParamSet2 = "Prob21: " & Format(MMECOAlpha21, "0.000") & ", Prob12: " & Format(MMECOAlpha12, "0.000")
ParamSet3 = "IDC: " & Format(MMECOCInf, "##0.##")


' Cumulative steady-state prob vector
SteadyCumVector(1) = SteadyVector(1)
For i = 2 To UBound(SteadyVector)
    SteadyCumVector(i) = SteadyVector(i) + SteadyCumVector(i - 1)
Next i


End Sub
Private Sub NSNPSim_BeginModel()

    Dim i, j, CurNode As Integer
    Dim sheetNext As Worksheet
    Dim CurString As String
    Dim found As Boolean
'    Dim LastNumNodes As Integer

    ReDim NodeSum(NumberOfSteps) As Double
    ReDim NodeSqSum(NumberOfSteps) As Double

    If Worksheets("Output").Visible = False Then
        Worksheets("Output").Visible = True
    End If
    If Worksheets("Arrivals").Visible = False Then
        Worksheets("Arrivals").Visible = True
    End If

    Simulation_StartClock = Now

    Sheets("Progress").Select
    Range("B17:F25").Select
    Selection.ClearContents
    Range("B18").Select

    Sheets("Output").Select
    Range("A2:FZ65536").Select
    Selection.ClearContents

    Sheets("Arrivals").Select
    Range("A1:IV65536").Select
    Selection.ClearContents
    Range("A1").Select
    Selection.value = "Rep"
```

```
    Range("B1").Select
    Selection.value = "Arrivals"

    MaxInvTime = 0

    Worksheets("Progress").Cells(17, 2).value = "Reps Done"
    Range("A1").Select

    ' Write Model
    With Worksheets("Progress")
        .Cells(18, 6).value = PhTypeString
        .Cells(19, 6).value = ParamSet1
        .Cells(20, 6).value = ParamSet2
        .Cells(21, 6).value = ParamSet3
    End With

    'ReDim NodeTrace(400, 4, NumNodes, Number_Reps) As Double
    'ReDim NodeTrace(800, 3, Number_Reps) As Double
    ReDim BigTable(NumberOfSteps, MaxMult) As Double
    'ReDim NodeCount As Integer                  ' incremented every time # of custs at each node chan
ges
    ReDim NodeLength(Number_Reps) As Integer                ' # of counts for each rep and for each no
de

    ReDim NodeSum(NumberOfSteps) As Double
    ReDim NodeSqSum(NumberOfSteps) As Double
    ReDim EN1st(NumberOfSteps) As Double
    ReDim EN2nd(NumberOfSteps) As Double
    ReDim VarN(NumberOfSteps) As Double
    ReDim RelErrorNum(NumberOfSteps) As Double
    ReDim VarMeanRat(NumberOfSteps) As Double
    ReDim RateFuncValue(NumberOfSteps) As Double
    ReDim DerivApprox(NumberOfSteps) As Double

    ReDim XVarRec(Number_Reps) As Double

    Sheets("Simulation Parameters").Select
    Range("A1").Select
End Sub

Private Sub StartRep(RepNumber As Integer)

    Dim CurNode As Integer
    Dim WhichBasePh As Double
    Dim R, CurPhase As Integer
    Dim FirstArrTime As Double
    Dim InvFirstTime As Double

    ReDim NodeTrace(20000, 3) As Double
    NodeCount = 0
    NodeTrace(1, 1) = NodeCount    ' step 1: step = 1
    NodeTrace(1, 2) = 0             ' step 1: time = 0
    NodeTrace(1, 3) = 0             ' step 1: # of arrivals by this time

    LastS = 0
    XVar = 0
    If WithDSPP = True Then
        If GammaN > 25 Then
            XVar = (1 / GammaN) * GammaSpecOne(GammaN, ExponStream)
        Else
            XVar = (1 / GammaN) * ErlangOneStep(GammaN, 1, ExponStream)
        End If
    Else
        XVar = 1
    End If

    LastS = XVar * MyCumRateFunc(Initial_SimTime)

    ''' Schedule ALL events
    'Call NewRead_Inputs(True, sim_time)

    ' Schedule first potential arrival
    If MajPhType = 1 Then               ' exponential
```

```
    FirstArrTime = Expon(ExponMean, ExponStream)

ElseIf MajPhType = 2 Then           ' h2b
    WhichBasePh = lcgrand(CompMixProb)
    If WhichBasePh < SteadyCumVector(1) Then
        ' initially in phase 1--generate exponential from lambda1
        FirstArrTime = Expon(1 / h2bRate, ExponStream)
    Else
        ' initially in phase 2--generate exponential from lambda2
        FirstArrTime = Expon(1 / (h2bRate * (1 - h2bAlpha) / h2bAlpha), ExponStream)
    End If

ElseIf MajPhType = 3 Then           ' MECon
    WhichBasePh = lcgrand(CompMixProb)
    R = 1
    Do Until WhichBasePh < SteadyCumVector(R)
        R = R + 1
        Loop
    CurPhase = R
    If CurPhase < MEConOrder + 1 Then
        ' initially in chain 1--generate Erlang of K+1-R
        FirstArrTime = ErlangOneStep(MEConOrder + 1 - CurPhase, 1 / MEConRate, ExponStream)
    Else
        ' initially in chain 2--generate Erlang of 2*K-R
        FirstArrTime = ErlangOneStep(2 * MEConOrder - CurPhase, 1 / MEConRate, ExponStream)
    End If

ElseIf MajPhType = 4 Then           ' MECO (3-moment)
    WhichBasePh = lcgrand(CompMixProb)
    R = 1
    Do Until WhichBasePh < SteadyCumVector(R)
        R = R + 1
        Loop
    CurPhase = R
    If CurPhase < MECOOrder + 1 Then
        ' initially in chain 1--generate Erlang of K+1-R, with rate1
        FirstArrTime = ErlangOneStep(MECOOrder + 1 - CurPhase, 1 / MECORate1, ExponStream)
    Else
        ' initially in chain 2--generate Erlang of 2*K+1-R, with rate2
        FirstArrTime = ErlangOneStep(2 * MECOOrder + 1 - CurPhase, 1 / MECORate2, ExponStream)
    End If

ElseIf MajPhType = 5 Then           ' Markov-MECO
    WhichBasePh = lcgrand(CompMixProb)
    R = 1
    Do Until WhichBasePh < SteadyCumVector(R)
        R = R + 1
        Loop
    CurPhase = R
    If CurPhase < MMECOOrder + 1 Then
        ' initially in chain 1--generate Erlang of K+1-R, with rate1
        FirstArrTime = ErlangOneStep(MMECOOrder + 1 - CurPhase, 1 / MMECORate1, ExponStream)
        PrevGenErlang = 1
    Else
        ' initially in chain 2--generate Erlang of 2*K+1-R, with rate2
        FirstArrTime = ErlangOneStep(2 * MMECOOrder + 1 - CurPhase, 1 / MMECORate2, ExponStream)
        PrevGenErlang = 2
    End If

End If

If FirstArrTime > LastS Then
    InvFirstTime = Initial_SimTime + 1
Else
    InvFirstTime = MyInvRateFunc(FirstArrTime / XVar)
End If

' First Arrival
Call event_schedule(sim_time + InvFirstTime, Event_Arrival)

'' First Update
'Call event_schedule(sim_time + StepSize, Event_UpdateNumbers)
```

```
    ' End Replication
    Call event_schedule(sim_time + Initial_SimTime, Event_End_Of_Simulation)


End Sub

Private Sub MakeArrival(PhType As Integer, RepNumber As Integer)

    Dim WhichBasePh As Double
    Dim AcceptOrNot As Double
    Dim InvTime As Double
    Dim NextArrTime As Double
    Dim CurArrTime As Double
    Dim Pi As Double

    'Pi = WorksheetFunction.Pi


    ' what to do with arrival
    ' Check to see if server idle
    If list_size(Serve_Check) < NumServ Then

        ' if yes, put into service
        Call event_schedule(sim_time + Expon(1 / ServRate, ServTime), Event_SvComp)
        Call list_file(LAST, Serve_Check)

    Else

        ' if no, put in queue
        Call list_file(LAST, Serve_Queue)

    End If
    ' make trace
    NodeCount = NodeCount + 1
    NodeTrace(NodeCount, 1) = NodeCount
    NodeTrace(NodeCount, 2) = sim_time
    NodeTrace(NodeCount, 3) = NodeTrace(NodeCount - 1, 3) + 1


    ' Schedule next potential arrival
    If MajPhType = 1 Then               ' exponential
        NextArrTime = Expon(ExponMean, ExponStream)

    ElseIf MajPhType = 2 Then           ' h2b
        WhichBasePh = lcgrand(CompMixProb)
        If WhichBasePh < h2bAlpha Then
            NextArrTime = Expon(1 / h2bRate, ExponStream)
        Else
            NextArrTime = Expon(1 / (h2bRate * (1 - h2bAlpha) / h2bAlpha), ExponStream)
        End If

    ElseIf MajPhType = 3 Then           ' MECon
        WhichBasePh = lcgrand(CompMixProb)
        If WhichBasePh < 1 - MEConAlpha Then
            NextArrTime = ErlangOneStep(MEConOrder, 1 / MEConRate, ExponStream)
        Else
            NextArrTime = ErlangOneStep(MEConOrder - 1, 1 / MEConRate, ExponStream)
        End If

    ElseIf MajPhType = 4 Then           ' MECO (3-moment)
        WhichBasePh = lcgrand(CompMixProb)
        If WhichBasePh < MECOAlpha Then
            NextArrTime = ErlangOneStep(MECOOrder, 1 / MECORate1, ExponStream)
        Else
            NextArrTime = ErlangOneStep(MECOOrder, 1 / MECORate2, ExponStream)
        End If

    ElseIf MajPhType = 5 Then           ' Markov-MECO
        WhichBasePh = lcgrand(CompMixProb)
        If PrevGenErlang = 1 Then
            If WhichBasePh < 1 - MMECOAlpha12 Then
                NextArrTime = ErlangOneStep(MMECOOrder, 1 / MMECORate1, ExponStream)
                PrevGenErlang = 1
```

```
            Else
                NextArrTime = ErlangOneStep(MMECOOrder, 1 / MMECORate2, ExponStream)
                PrevGenErlang = 2
            End If
        Else
            If WhichBasePh < MMECOAlpha21 Then
                NextArrTime = ErlangOneStep(MMECOOrder, 1 / MMECORate1, ExponStream)
                PrevGenErlang = 1
            Else
                NextArrTime = ErlangOneStep(MMECOOrder, 1 / MMECORate2, ExponStream)
                PrevGenErlang = 2
            End If
        End If

    End If
    ' Calculate NS arrival time: R^-1(NextArrTime)
    CurArrTime = XVar * MyCumRateFunc(sim_time)
    If CurArrTime + NextArrTime > LastS Then
        InvTime = Initial_SimTime + 1
    Else
        InvTime = MyInvRateFunc((CurArrTime + NextArrTime) / XVar)
        ' Schedule next arrival
        Call event_schedule(InvTime, Event_Arrival)
    End If

End Sub
Private Sub CompleteService(RepNumber As Integer)

Dim SvProbQ1 As Double
Dim InitSv1, InitSv2 As Double
Dim i, k, l As Integer

    ' Assume model is NSNP/M/1--will need to recode for NSNP/Ph/1

    ' complete current service
    Call list_remove(FIRST, Serve_Check)

    ' put first in queue into service, if necessary
    If list_size(Serve_Queue) = 0 Then
    Else
        ' put first in queue into service
        Call list_remove(FIRST, Serve_Queue)
        If ViewThrough = True Then
            Call event_schedule(sim_time + Expon(1 / ServRate, ServTime), Event_SvComp)
        Else
            Call event_schedule(sim_time + 2 * Initial_SimTime, Event_SvComp)
        End If
        Call list_file(LAST, Serve_Check)
    End If

    ' make trace
    If ViewThrough = True Then
        NodeCount = NodeCount + 1
        NodeTrace(NodeCount, 1) = NodeCount
        NodeTrace(NodeCount, 2) = sim_time
        NodeTrace(NodeCount, 3) = NodeTrace(NodeCount - 1, 3) - 1
    End If


End Sub


Private Sub End_Simulation(RepNumber As Integer)

    Dim CurNode As Integer
    Dim i, j, k, l, R, s, t As Integer
    Dim CurrentTime As Double
    Dim Temp As Double
    Dim UseK As Integer
    Dim CurPt, DepCurPt As Integer
    Dim TempCusts As Integer
    Dim TempDepCounts As Integer
```

```
    NodeLength(RepNumber) = NodeCount
    For k = 1 To NodeLength(RepNumber)
        TotalNodeTraces(k, RepNumber) = NodeTrace(k, 2)
    Next k

    For i = 1 To NumberOfSteps

        CurrentTime = (i - 1) * StepSize

        k = 1
        Do Until NodeTrace(k, 2) > CurrentTime
            k = k + 1
            If k > NodeLength(RepNumber) Then
                'k = k - 1
                Exit Do
            End If
            Loop

        UseK = k - 1

        If i > 1 Then
            If NodeTrace(UseK, 2) < CurrentTime Then
                TempCusts = NodeTrace(UseK, 3)
                NodeSum(i) = NodeSum(i) + TempCusts
                NodeSqSum(i) = NodeSqSum(i) + TempCusts ^ 2
            End If
        End If

    Next i

    ' Show Rep Count
    If RepNumber = IntNote * Int(RepNumber / IntNote) Then
        Worksheets("Progress").Select
        Worksheets("Progress").Cells(18, 2).value = RepNumber
        Range("B18").Select
    End If

    '' Show Example Traces at Each Node
    If Number_Reps < 50001 Then
        If RepNumber = CurMult * Int(Number_Reps / MaxMult) Then
            Worksheets("Arrivals").Cells(1, 5 + (CurMult - 1) * 6).value = "Rep " & Format(RepNumber,
"######")
            For CurPt = 1 To NodeCount
                For i = 1 To 3
                    Worksheets("Arrivals").Cells(1 + CurPt, (i - 1) + 5 + (CurMult - 1) * 6).value = N
odeTrace(CurPt, i)
                Next i
            Next CurPt
            CurMult = CurMult + 1

            For i = 1 To NumberOfSteps

                CurrentTime = (i - 1) * StepSize

                k = 1
                Do Until NodeTrace(k, 2) > CurrentTime
                    k = k + 1
                    If k > NodeLength(RepNumber) Then
                        'k = k - 1
                        Exit Do
                    End If
                    Loop

                UseK = k - 1

                If i > 1 Then
                    If NodeTrace(UseK, 2) < CurrentTime Then
                        TempCusts = NodeTrace(UseK, 3)
                        BigTable(i, CurMult - 1) = TempCusts
                    End If
                End If
```

```
            Next i



        End If
    End If

    XVarRec(RepNumber) = XVar


End Sub

Private Sub GetResults()

    Dim i, j, k, l, R, s, t As Integer
    'Dim NodeSum() As Double
    'Dim NodeSqSum() As Double
    'Dim NodeDepsSum() As Double
    'Dim NodeDepsSqSum() As Double
    Dim CurrentTime As Double
    Dim Temp As Double
    Dim CurNode As Integer
    Dim UseK As Integer
    Dim DepCurPt As Integer
    Dim TempCusts As Integer
    Dim TempDepCounts As Integer
    Dim MaxRepLength As Integer

    Dim Sim_RunTime As Double
    Dim MinNumSteps As Integer
    Dim CurMultPiece As Integer
    Dim CalcEndTime, CalcRunTime As Double

    Dim VarMeanTitle As String

    Worksheets("Output").Cells(2, 2).value = "Time"
    Worksheets("Output").Cells(2, 6).value = "Time"
    Worksheets("Output").Cells(2, 10).value = "Time"
    Worksheets("Output").Cells(2, 14).value = "Time"
    Worksheets("Output").Cells(2, 18).value = "Time"
    Worksheets("Output").Cells(2, 22).value = "Time"
    Worksheets("Output").Cells(2, 3).value = "SimMean"
    Worksheets("Output").Cells(2, 4).value = "MDEMean"
    Worksheets("Output").Cells(2, 7).value = "SimVar"
    Worksheets("Output").Cells(2, 8).value = "MDEVar"
    Worksheets("Output").Cells(2, 11).value = "Rel Error"
    Worksheets("Output").Cells(2, 15).value = "Var-Mean Ratio"
    Worksheets("Output").Cells(2, 19).value = "r(t)"
    Worksheets("Output").Cells(2, 20).value = "Deriv of EN"

    MaxRepLength = 0

    ' write out number of arrivals in each rep
    For j = 1 To Number_Reps
        Worksheets("Arrivals").Cells(j + 1, 1).value = j
        Worksheets("Arrivals").Cells(j + 1, 2).value = NodeLength(j)
        Worksheets("Arrivals").Cells(j + 1, 3).value = XVarRec(j)
        If NodeLength(j) > MaxRepLength Then
            MaxRepLength = NodeLength(j)
        End If
    Next j
    Worksheets("Arrivals").Cells(1, 3).FormulaR1C1 = "=MAX(R[1]C[-1]:R[65535]C[-1])"

    For i = 1 To NumberOfSteps

        Worksheets("Output").Cells(i + 2, 2).value = (i - 1) * StepSize

        ' get average
        EN1st(i) = NodeSum(i) / Number_Reps
        VarN(i) = (NodeSqSum(i) - Number_Reps * (EN1st(i) ^ 2)) / (Number_Reps - 1)
        If EN1st(i) > 0 Then
            RelErrorNum(i) = 1.96 * VBA.Sqr(VarN(i) / Number_Reps) / EN1st(i)
            VarMeanRat(i) = VarN(i) / EN1st(i)
```

```
        Else
            RelErrorNum(i) = 0
            If VarN(i) = 0 Then
                VarMeanRat(i) = 1
            Else
                VarMeanRat(i) = 0
            End If
        End If
        ' output to worksheet
        Worksheets("Output").Cells(i + 2, 2).value = StepSize * (i - 1)
        Worksheets("Output").Cells(i + 2, 6).value = StepSize * (i - 1)
        Worksheets("Output").Cells(i + 2, 10).value = StepSize * (i - 1)
        Worksheets("Output").Cells(i + 2, 14).value = StepSize * (i - 1)
        Worksheets("Output").Cells(i + 2, 18).value = StepSize * (i - 1)
        Worksheets("Output").Cells(i + 2, 3).value = EN1st(i)
        Worksheets("Output").Cells(i + 2, 7).value = VarN(i)
        Worksheets("Output").Cells(i + 2, 11).value = RelErrorNum(i)
        Worksheets("Output").Cells(i + 2, 11).Style = "Percent"
        Worksheets("Output").Cells(i + 2, 11).NumberFormat = "0.00%"
        Worksheets("Output").Cells(i + 2, 15).value = VarMeanRat(i)

    Next i

    'delete old sheets
    Application.DisplayAlerts = False
    Sheets(Array("ENt Graph", "VNt Graph", "RelError Graph", "V-M Ratio Graph", "SamplePaths")).Select
    Sheets("ENt Graph").Activate
    ActiveWindow.SelectedSheets.Delete
    Application.DisplayAlerts = True

    If WithDSPP = True Then
        VarMeanTitle = "GammaN = " & Format(GammaN, "##,##0")
    Else
        VarMeanTitle = "scv(X) = " & Format(scvX, "##0.0##") & ", rho1(x) = " & Format(rho1X, "0.0##")
    End If

    ' make chart for EN
    Sheets("Output").Select
    Range("C2").Select
    'Range(Selection, Selection.End(xlToRight)).Select
    Range(Selection, Selection.End(xlDown)).Select
    Charts.Add
    ActiveChart.ChartType = xlLineMarkers
    ActiveChart.SetSourceData Source:=Sheets("Output").Range("C2:D" & Format(2 + NumberOfSteps, "#####
")), PlotBy _
        :=xlColumns
    ActiveChart.SeriesCollection(1).XValues = "=Output!R3C2:R" & Format(NumberOfSteps + 2, "#####") &
"C2"
    'ActiveChart.SeriesCollection(1).Name = "=Output!R2C3"
    'ActiveChart.SeriesCollection(2).Name = "=Output!R2C4"
    ActiveChart.Location Where:=xlLocationAsNewSheet, Name:="ENt Graph"
    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "Mean of NS Arrival Count, Inversion Method, " & VarMeanTitle
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "time"
        .Axes(xlValue, xlPrimary).HasTitle = False
    End With
    ActiveChart.HasLegend = False
    Call FixPlotPiecesForNice_PlotArea
    Call FixPlotPiecesForNice_yPretty
    ActiveChart.Deselect


    ' make chart for VN
    'Sheets("VNt Graph").Select
    'ActiveWindow.SelectedSheets.Delete
    Sheets("Output").Select
    Range("F2").Select
    'Range(Selection, Selection.End(xlToRight)).Select
    Range(Selection, Selection.End(xlDown)).Select
    Charts.Add
    ActiveChart.ChartType = xlLineMarkers
```

```
    ActiveChart.SetSourceData Source:=Sheets("Output").Range("G2:H" & Format(2 + NumberOfSteps, "#####
")), PlotBy _
        :=xlColumns
    ActiveChart.SeriesCollection(1).XValues = "=Output!R3C6:R" & Format(NumberOfSteps + 2, "#####") &
"C6"
    'ActiveChart.SeriesCollection(1).Name = "=Output!R2C7"
    'ActiveChart.SeriesCollection(2).Name = "=Output!R2C8"
    ActiveChart.Location Where:=xlLocationAsNewSheet, Name:="VNt Graph"
    With ActiveChart
        .HasTitle = True
        '.ChartTitle.Characters.Text = "Variance of Size, NSNP/M/1 Queue, Inversion Method, " & VarMea
nTitle
        .ChartTitle.Characters.Text = "Variance of NS Arrival Count, Inversion Method, " & VarMeanTitl
e
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "time"
        .Axes(xlValue, xlPrimary).HasTitle = False
    End With
    ActiveChart.HasLegend = False
    Call FixPlotPiecesForNice_PlotArea
    Call FixPlotPiecesForNice_yPretty
    ActiveChart.Deselect

    ' make chart for RelError
    Sheets("Output").Select
    Range("I2").Select
    'Range(Selection, Selection.End(xlToRight)).Select
    Range(Selection, Selection.End(xlDown)).Select
    Charts.Add
    ActiveChart.ChartType = xlLineMarkers
    ActiveChart.SetSourceData Source:=Sheets("Output").Range("K2:K" & Format(2 + NumberOfSteps, "#####
")), PlotBy _
        :=xlColumns
    ActiveChart.SeriesCollection(1).XValues = "=Output!R3C10:R" & Format(NumberOfSteps + 2, "#####") &
 "C10"
    ActiveChart.Location Where:=xlLocationAsNewSheet, Name:="RelError Graph"
    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "Relative Error for Mean Estimate, Simulation of Inversion Metho
d"
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "time"
        .Axes(xlValue, xlPrimary).HasTitle = False
    End With
    ActiveChart.HasLegend = False
    Call FixPlotPiecesForNice_PlotArea
    ActiveChart.Deselect

    ' make chart for Var-to-Mean Ratio
    Sheets("Output").Select
    Range("L2").Select
    'Range(Selection, Selection.End(xlToRight)).Select
    'Range(Selection, Selection.End(xlDown)).Select
    Charts.Add
    ActiveChart.ChartType = xlLineMarkers
    ActiveChart.SetSourceData Source:=Sheets("Output").Range("O2:O" & Format(2 + NumberOfSteps, "#####
")), PlotBy _
        :=xlColumns
    ActiveChart.SeriesCollection(1).XValues = "=Output!R3C14:R" & Format(NumberOfSteps + 2, "#####") &
 "C14"
    ActiveChart.Location Where:=xlLocationAsNewSheet, Name:="V-M Ratio Graph"
    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "Variance-to-Mean Ratio, Cumulative Arrival Count, " & VarMeanTi
tle & ", (IDC = " & Format(MMECOCInf, "##0.0##") & ")"
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "time"
        .Axes(xlValue, xlPrimary).HasTitle = False
    End With
    ActiveChart.HasLegend = False
    Call FixPlotPiecesForNice_PlotArea
    ActiveChart.Deselect
```

```
    '' make chart for Rate-pieces Ratio
    'Sheets("Output").Select
    'Range("O2").Select
    'Range(Selection, Selection.End(xlToRight)).Select
    'Range(Selection, Selection.End(xlDown)).Select
    'Charts.Add
    'ActiveChart.ChartType = xlLineMarkers
    'ActiveChart.SetSourceData Source:=Sheets("Output").Range("O2:P" & Format(2 + MinNumSteps, "#####"
)), PlotBy _
    '    :=xlColumns
    'ActiveChart.SeriesCollection(1).XValues = "=Output!R3C14:R" & Format(MinNumSteps + 2, "#####") &
"C14"
    'ActiveChart.Location Where:=xlLocationAsNewSheet, Name:="Captured Rate Graph"
    'With ActiveChart
    '    .HasTitle = True
    '    .ChartTitle.Characters.Text = "r(t) vs. Change in Mean Arrival Count per Step"
    '    .Axes(xlCategory, xlPrimary).HasTitle = True
    '    .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "time"
    '    .Axes(xlValue, xlPrimary).HasTitle = False
    'End With
    'ActiveChart.HasLegend = True
    'ActiveChart.Legend.Select
    'Selection.Position = xlBottom

    ' Make Graph for MaxMult sample paths
    ' Assume MaxMult = 40
    'ReDim BigTable(NumberOfSteps, MaxMult) As Double
    For j = 1 To MaxMult
    '    CurMultPiece = j * Number_Reps / MaxMult
    '
    '    Worksheets("Output").Cells(3, 18 + j).value = 0
        For i = 1 To NumberOfSteps
    '
    '        CurrentTime = (i - 1) * StepSize
    '        ' output to worksheet
    '        If j = 1 Then
    '            Worksheets("Output").Cells(i + 2, 18).value = CurrentTime
    '        End If
    '
    '        k = 1
    '        Do Until NodeTrace(k, 2, CurMultPiece) > CurrentTime
    '            k = k + 1
    '            If k > NodeLength(CurMultPiece) Then
    '                'k = k - 1
    '                Exit Do
    '            End If
    '            Loop
    '
    '       UseK = k - 1
    '
    '        If i > 1 Then
    '            If NodeTrace(UseK, 2, CurMultPiece) < CurrentTime Then
    '                BigTable(i, j) = NodeTrace(UseK, 3, CurMultPiece)
                   Worksheets("Output").Cells(i + 2, 18 + j).value = BigTable(i, j)
    '            End If
    '        End If
    '
        Next i
    Next j
    '' make chart
    Sheets("Output").Select
    Range("R2").Select
    Range(Selection, Selection.End(xlToRight)).Select
    Range(Selection, Selection.End(xlDown)).Select
    Charts.Add
    ActiveChart.ChartType = xlLineMarkers
    If MaxMult = 10 Then
        ActiveChart.SetSourceData Source:=Sheets("Output").Range("S2:AB" & Format(2 + NumberOfSteps, "
#####")), PlotBy _
            :=xlColumns
    ElseIf MaxMult = 40 Then
        ActiveChart.SetSourceData Source:=Sheets("Output").Range("S2:BF" & Format(2 + NumberOfSteps, "
#####")), PlotBy _
```

```
            :=xlColumns
    End If
    ActiveChart.SeriesCollection(1).XValues = "=Output!R3C18:R" & Format(NumberOfSteps + 2, "#####") &
"C18"
    ActiveChart.Location Where:=xlLocationAsNewSheet, Name:="SamplePaths"
    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "Sample Paths (" & Format(MaxMult, "##,##0") & "), Inversion Met
hod, " & VarMeanTitle
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "time"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "I(t)"
    End With
    ActiveChart.HasLegend = False
    'ActiveChart.Legend.Select
    'Selection.Position = xlBottom
    ActiveChart.PlotArea.Select
    With Selection.Border
        .Weight = xlThin
        .LineStyle = xlNone
    End With
    Selection.Interior.ColorIndex = xlNone
    ActiveChart.Axes(xlCategory).Select
    With ActiveChart.Axes(xlCategory)
        .CrossesAt = 1
        .TickLabelSpacing = Int(NumberOfSteps / 10)
        .TickMarkSpacing = 1
        .AxisBetweenCategories = True
        .ReversePlotOrder = False
    End With
    ActiveChart.Axes(xlValue).Select
    Selection.TickLabels.NumberFormat = "#,##0"

    Call MakeSamplePathYGood
    ' clean all data series
    For j = 1 To MaxMult
        ActiveChart.SeriesCollection(j).Select
        With Selection.Border
            .ColorIndex = 57
            .Weight = xlMedium
            .LineStyle = xlContinuous
        End With
        With Selection
            .MarkerBackgroundColorIndex = xlNone
            .MarkerForegroundColorIndex = xlAutomatic
            .MarkerStyle = xlNone
            .Smooth = False
            .MarkerSize = 7
            .Shadow = False
        End With
    Next j
    ' fix sample path 10, which is baby blue
    ActiveChart.SeriesCollection(10).Select
    With Selection.Border
        .ColorIndex = 3
        .Weight = xlMedium
        .LineStyle = xlContinuous
    End With
    With Selection
        .MarkerBackgroundColorIndex = xlNone
        .MarkerForegroundColorIndex = xlNone
        .MarkerStyle = xlNone
        .Smooth = False
        .MarkerSize = 7
        .Shadow = False
    End With
    ActiveChart.Deselect

    ' output simulation length
    Worksheets("Progress").Cells(23, 2).value = "Calculation Time"
    CalcEndTime = Now
    CalcRunTime = CalcEndTime - Simulation_EndClock
```

```
    Worksheets("Progress").Cells(24, 2).value = CalcRunTime

    'Worksheets("Simulation Parameters").Select

End Sub
Private Sub FixPlotPiecesForNice_PlotArea()

' 1. make plot area white, clear border
ActiveChart.PlotArea.Select
With Selection.Border
    .Weight = xlThin
    .LineStyle = xlNone
End With
Selection.Interior.ColorIndex = xlNone
' 2. make plots thin, no markers
ActiveChart.SeriesCollection(1).Select
With Selection.Border
    .Weight = xlThin
    .LineStyle = xlAutomatic
End With
With Selection
    .MarkerBackgroundColorIndex = xlAutomatic
    .MarkerForegroundColorIndex = xlAutomatic
    .MarkerStyle = xlNone
    .Smooth = False
    .MarkerSize = 5
    .Shadow = False
End With
With Selection.Border
    .ColorIndex = 57
    .Weight = xlThick
    .LineStyle = xlContinuous
End With
With Selection
    .MarkerBackgroundColorIndex = xlNone
    .MarkerForegroundColorIndex = xlNone
    .MarkerStyle = xlNone
    .Smooth = False
    .MarkerSize = 5
    .Shadow = False
End With
' 3. space out timelabels on x-axis
ActiveChart.Axes(xlCategory).Select
With ActiveChart.Axes(xlCategory)
    .CrossesAt = 1
    .TickLabelSpacing = Int(NumberOfSteps / 10)
    .TickMarkSpacing = 1
    .AxisBetweenCategories = True
    .ReversePlotOrder = False
End With

End Sub

Private Sub FixPlotPiecesForNice_yPretty()
' 4. make y-axis title pretty
ActiveChart.Axes(xlValue).Select
Selection.TickLabels.NumberFormat = "#,##0"

End Sub
Sub MakeSamplePathYGood()


ActiveChart.Axes(xlValue).AxisTitle.Select
Selection.Characters.Text = "I(t)"
Selection.AutoScaleFont = False
With Selection.Characters(Start:=1, Length:=1).Font
    .Name = "Book Antiqua"
    .FontStyle = "Bold Italic"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
```

```
    .Shadow = False
    .Underline = xlUnderlineStyleNone
    .ColorIndex = xlAutomatic
End With
Selection.AutoScaleFont = False
With Selection.Characters(Start:=2, Length:=1).Font
    .Name = "Arial"
    .FontStyle = "Bold"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlUnderlineStyleNone
    .ColorIndex = xlAutomatic
End With
Selection.AutoScaleFont = False
With Selection.Characters(Start:=3, Length:=1).Font
    .Name = "Arial"
    .FontStyle = "Bold Italic"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlUnderlineStyleNone
    .ColorIndex = xlAutomatic
End With
Selection.AutoScaleFont = False
With Selection.Characters(Start:=4, Length:=1).Font
    .Name = "Arial"
    .FontStyle = "Bold"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlUnderlineStyleNone
    .ColorIndex = xlAutomatic
End With

End Sub
Public Sub GetRho1Range()

' based on scv, get feasible range of rho1
Dim i, j, k, R, s As Integer
Dim l As Long
Dim CurNode As Integer

Dim ArrString As String
''Dim SvString() As Variant
'Dim MeanSvTime As Double
''Dim SCV As Double
' For MECO
Dim ProbMoms
Dim MinN1, MinN2 As Double
Dim mecoA, mecoB, mecoC, mecoX, mecoY As Double
Dim mecoM1, mecoM2, mecoM3 As Double
Dim mecoRoot1, mecoRoot2 As Double
' To get Max r(t)
Dim CurVal As Double
Dim CurMax As Double
Dim TheTime As Double
Dim TempMax As Double
Dim LargeNumSteps As Long
Dim SmallIntDiv As Integer

' new terms for Markov-MECO
Dim TwoMomsOrder As Integer
Dim TMat() As Double        ' this is D0 \equiv U(A1-I)
```

```
Dim TMatInv() As Double
Dim TMatSqInv() As Double
Dim TMatCuInv() As Double
Dim ZetaVec() As Double     ' this is zeta vec
Dim TrueVarMoms As Double   ' this is true var
Dim TrueSecMoms As Double   ' this is true m2
Dim ImpThirdMoms As Double  ' this is implied m3
Dim SkewX As Double         ' this is standardized third moment
Dim Coeffvar As Double      ' this is cv = sqr(scvx)
Dim NeedBiggerOrder As Boolean
Dim tempMMECOCov As Double

' new terms for finding min and max rho1
Dim MinRho1 As Double
Dim MaxRho1 As Double
Dim PrevMinGuess As Double
Dim CurMinGuess As Double
Dim PrevMaxGuess As Double
Dim CurMaxGuess As Double
Dim InitK As Integer
Dim CountMin As Integer
Dim CountMax As Integer
Dim FoundMin As Boolean
Dim FoundMax As Boolean
Dim PrintRhoString As String
Dim DispRhoString
Dim MaxOrderSize As Long
Dim TempMECOOrder As Long

MaxOrderSize = 50000
CanMakeMMECO = True

' Simulation Parameters
scvX = Worksheets("Simulation Parameters").Range("NSNR.TargSCV").value

MeanSvTime = 1
TrueVarMoms = scvX * MeanSvTime ^ 2
TrueSecMoms = TrueVarMoms + MeanSvTime ^ 2
Coeffvar = VBA.Sqr(scvX)
' Get Implied Third Moment
If scvX = 1 Then
    ' exponential
    TwoMomsOrder = 1
    ImpThirdMoms = 6

ElseIf scvX < 1 Then
    ' MECon

    ' determine K
    k = 1
    Do While 1 / k > scvX
        k = k + 1
        Loop
    ' Parameters
    MEConOrder = k
    MEConAlpha = (1 / (1 + scvX)) * (k * scvX - VBA.Sqr(k * (1 + scvX) - k ^ 2 * scvX))
    MEConRate = (k - MEConAlpha) / MeanSvTime

    TwoMomsOrder = 2 * MEConOrder - 1
    ReDim TMat(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double       ' this is D0 \equiv U(A1-I)
    ReDim TMatInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatSqInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatCuInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim ZetaVec(1 To TwoMomsOrder) As Double     ' this is zeta vec
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            TMat(i, j) = 0
            TMatInv(i, j) = 0
            TMatSqInv(i, j) = 0
            TMatCuInv(i, j) = 0
        Next j
        ZetaVec(i) = 0
    Next i
```

```
    ZetaVec(1) = 1 - MEConAlpha
    ZetaVec(MEConOrder + 1) = MEConAlpha
    ' TMatInv directly: relative easy
    For i = 1 To MEConOrder
        For j = i To MEConOrder
            TMatInv(i, j) = -1 / MEConRate
        Next j
    Next i
    For i = MEConOrder + 1 To 2 * MEConOrder - 1
        For j = i To 2 * MEConOrder - 1
            TMatInv(i, j) = -1 / MEConRate
        Next j
    Next i

Else
    ' h2b
    TwoMomsOrder = 2
    ReDim TMat(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double       ' this is D0 \equiv U(A1-I)
    ReDim TMatInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatSqInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim TMatCuInv(1 To TwoMomsOrder, 1 To TwoMomsOrder) As Double
    ReDim ZetaVec(1 To TwoMomsOrder) As Double     ' this is zeta vec
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            TMat(i, j) = 0
            TMatInv(i, j) = 0
            TMatSqInv(i, j) = 0
            TMatCuInv(i, j) = 0
        Next j
        ZetaVec(i) = 0
    Next i

    ' get h2b params
    ' Parameters
    h2bAlpha = 0.5 * (1 + VBA.Sqr(1 - 2 / (scvX + 1)))
    h2bRate = 2 * h2bAlpha / MeanSvTime

    ' build T, zeta
    TMat(1, 1) = -h2bRate
    TMat(2, 2) = -h2bRate * (1 - h2bAlpha) / h2bAlpha
    ZetaVec(1) = h2bAlpha
    ZetaVec(2) = 1 - h2bAlpha
    ' invT for h2b is easy b/c T is diagonal
    For i = 1 To TwoMomsOrder
        TMatInv(i, i) = 1 / TMat(i, i)
    Next i

End If

If scvX > 1 Or scvX < 1 Then
    ' calc implied third moment
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            For k = 1 To TwoMomsOrder
                TMatSqInv(i, j) = TMatSqInv(i, j) + TMatInv(i, k) * TMatInv(k, j)
            Next k
        Next j
    Next i
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            For k = 1 To TwoMomsOrder
                TMatCuInv(i, j) = TMatCuInv(i, j) + TMatInv(i, k) * TMatSqInv(k, j)
            Next k
        Next j
    Next i
    For i = 1 To TwoMomsOrder
        For j = 1 To TwoMomsOrder
            ImpThirdMoms = ImpThirdMoms - 6 * ZetaVec(i) * TMatCuInv(i, j)
        Next j
    Next i
End If

' find minimum Markov-MECO order
```

```
SkewX = (ImpThirdMoms - 3 * MeanSvTime * TrueSecMoms + 2 * MeanSvTime ^ 3) / (TrueVarMoms ^ (3 / 2))

' Specfy Markov-MECO to match
If Coeffvar > 0 And SkewX >= Coeffvar - 1 / Coeffvar Then
    MinN1 = 1 / scvX
    MinN2 = (-SkewX + 1 / (Coeffvar ^ 3) + 1 / Coeffvar + 2 * Coeffvar) / (SkewX - (Coeffvar - 1 / Coe
ffvar))

    ' Check if given order feasible
    If MinN1 > MinN2 Then
        k = Int(MinN1) + 1
    Else
        k = Int(MinN2) + 1
    End If
Else
    ProbMoms = VBA.MsgBox("Bad MECO", vbOKCancel + vbCritical)
    CanMakeMMECO = False
    Exit Sub
End If

If MinN1 > MinN2 Then
    InitK = Int(MinN1) + 1
Else
    InitK = Int(MinN2) + 1
End If

CurMaxGuess = 1
FoundMax = False

' start looking for max rho1 achievable
k = InitK
Do While FoundMax = False

    NeedBiggerOrder = True
    Do While NeedBiggerOrder = True
        mecoM1 = MeanSvTime
        mecoM2 = TrueSecMoms
        mecoM3 = ImpThirdMoms

        mecoX = mecoM1 * mecoM3 - ((k + 2) / (k + 1)) * mecoM2 ^ 2
        mecoY = mecoM2 - ((k + 1) / k) * mecoM1 ^ 2
        mecoA = k * (k + 2) * mecoM1 * mecoY
        mecoB = -(k * mecoX + k * ((k + 2) / (k + 1)) * mecoY ^ 2 + (k + 2) * mecoY * mecoM1 ^ 2)
        mecoC = mecoM1 * mecoX

        mecoRoot1 = (-mecoB + VBA.Sqr(mecoB ^ 2 - 4 * mecoA * mecoC)) / (2 * mecoA)
        mecoRoot2 = (-mecoB - VBA.Sqr(mecoB ^ 2 - 4 * mecoA * mecoC)) / (2 * mecoA)

        TempMECOOrder = k
        MECORate1 = 1 / mecoRoot1
        MECORate2 = 1 / mecoRoot2
        MECOAlpha = ((mecoM1 / TempMECOOrder) - mecoRoot2) / (mecoRoot1 - mecoRoot2)

        ' now get Markov-MECO stuff
        tempMMECOCov = TrueVarMoms * CurMaxGuess
        MMECOAlpha21 = MECOAlpha - tempMMECOCov / ((1 - MECOAlpha) * (TempMECOOrder * mecoRoot1 - Temp
MECOOrder * mecoRoot2) ^ 2)
        MMECOAlpha12 = MMECOAlpha21 * (1 - MECOAlpha) / MECOAlpha
        MMECORate1 = MECORate1
        MMECORate2 = MECORate2
        ' check feasibility of M-MECO probs
        If MMECOAlpha21 < 0 Or MMECOAlpha21 > 1 Or MMECOAlpha12 < 0 Or MMECOAlpha12 > 1 Then
            k = k + 1
            If k > MaxOrderSize Then
                NeedBiggerOrder = False
                CurMaxGuess = CurMaxGuess - 0.001
                k = InitK
            End If

        Else
            'MMECOOrder = MECOOrder
            NeedBiggerOrder = False
            FoundMax = True
```

```
MSNPSim - 24
        End If

        Loop
    Loop
MaxRho1 = CurMaxGuess

' start looking for min rho1 achievable
CurMinGuess = MaxRho1
FoundMin = False
k = InitK
Do While FoundMin = False

    NeedBiggerOrder = True
    Do While NeedBiggerOrder = True
        mecoM1 = MeanSvTime
        mecoM2 = TrueSecMoms
        mecoM3 = ImpThirdMoms

        mecoX = mecoM1 * mecoM3 - ((k + 2) / (k + 1)) * mecoM2 ^ 2
        mecoY = mecoM2 - ((k + 1) / k) * mecoM1 ^ 2
        mecoA = k * (k + 2) * mecoM1 * mecoY
        mecoB = -(k * mecoX + k * ((k + 2) / (k + 1)) * mecoY ^ 2 + (k + 2) * mecoY * mecoM1 ^ 2)
        mecoC = mecoM1 * mecoX

        mecoRoot1 = (-mecoB + VBA.Sqr(mecoB ^ 2 - 4 * mecoA * mecoC)) / (2 * mecoA)
        mecoRoot2 = (-mecoB - VBA.Sqr(mecoB ^ 2 - 4 * mecoA * mecoC)) / (2 * mecoA)

        TempMECOOrder = k
        MECORate1 = 1 / mecoRoot1
        MECORate2 = 1 / mecoRoot2
        MECOAlpha = ((mecoM1 / TempMECOOrder) - mecoRoot2) / (mecoRoot1 - mecoRoot2)

        ' now get Markov-MECO stuff
        tempMMECOCov = TrueVarMoms * CurMinGuess
        MMECOAlpha21 = MECOAlpha - tempMMECOCov / ((1 - MECOAlpha) * (TempMECOOrder * mecoRoot1 - Temp
MECOOrder * mecoRoot2) ^ 2)
        MMECOAlpha12 = MMECOAlpha21 * (1 - MECOAlpha) / MECOAlpha
        MMECORate1 = MECORate1
        MMECORate2 = MECORate2
        ' check feasibility of M-MECO probs
        If MMECOAlpha21 < 0 Or MMECOAlpha21 > 1 Or MMECOAlpha12 < 0 Or MMECOAlpha12 > 1 Then
            k = k + 1
            If k > MaxOrderSize Then
                'ProbMoms = VBA.MsgBox("Cannot match target moments.", vbCritical)
                'CanMakeMMECO = False
                'Stop
                'Exit Sub
                FoundMin = True
                NeedBiggerOrder = False
            End If

        Else
            NeedBiggerOrder = False
            CurMinGuess = CurMinGuess - 0.001
            k = InitK
            'MMECOOrder = MECOOrder
        End If

        Loop
    Loop
MinRho1 = CurMinGuess

' Model String
PrintRhoString = "Feasible rho1 in [" & Format(MinRho1, "0.###") & "," & Format(MaxRho1, "0.###") & "]
"
'Worksheets("Simulation Parameters").Cells(17, 3).value = PrintRhoString
DispRhoString = VBA.MsgBox(PrintRhoString, vbOKOnly)
'Exit Sub

End Sub

Private Sub OutputAllReps()
```

→ Esto lo imprime en la página del excel.

```
Dim SheetNum As Integer
Dim MaxSheetCount As Integer
Dim RepsPerSheet As Integer
Dim RepThresh As Integer
Dim CurRep As Integer
Dim CurFileNameString As String
Dim ModelDescString As String
Dim ThisTimeString As String
Dim NewFileNameString As String
Dim scvString As String
Dim rho1String As String
Dim ArrCount As Integer
Dim CurSheetName As String
Dim CurrDirectoryString As String
Dim CurSheetNum As Integer
Dim ArrTimeString As String

''' hides Output, Arrivals worksheets
Worksheets("Output").Visible = False
Worksheets("Arrivals").Visible = False

''' return current focus to "Simulation Parameters" sheet
Worksheets("Simulation Parameters").Select

RepsPerSheet = 250
RepThresh = 0

If OutputArrivalTimes = True Then
    ArrTimeString = "ArrTimes_"
Else
    ArrTimeString = "InterarrTimes_"
End If
scvString = "scv(" & Format(scvX, "####0.0##") & ")"
'scvString = "scv(" & Format(Int(scvX), "##0") & "." & Format(1000 * (scvX - Int(scvX)), "0##") & ")"
If rho1X < 0 Then
    rho1String = "rho1(-" & Format(Abs(rho1X), "0.0##") & ")"
Else
    rho1String = "rho1(" & Format(rho1X, "0.0##") & ")"
End If

ModelDescString = ArrTimeString & scvString & rho1String & "EndTime(" & Format(Initial_SimTime, "###,##
#0") & ")"
''' build new workbook

CurFileNameString = ActiveWorkbook.Name
ThisTimeString = Format(Now, "mmddyyyy_hhmmss")
'CurrDirectoryString = "C:\Documents and Settings\Ira Gerhardt\My Documents\IFG NU Files\Research\Spri
ng '09 - Multi Finite Server Networks\Excel Files"
CurrDirectoryString = CurDir

Workbooks.Add
Sheets("Sheet1").Select
For CurRep = 1 To Number_Reps

    With ActiveSheet
        .Cells(1, CurRep - RepThresh).value = "Rep " & Format(CurRep, "####0")
        For ArrCount = 1 To NodeLength(CurRep)
            If OutputArrivalTimes = True Then
                .Cells(ArrCount + 1, CurRep - RepThresh).value = TotalNodeTraces(ArrCount, CurRep)
            Else
                If ArrCount = 1 Then
                    .Cells(ArrCount + 1, CurRep - RepThresh).value = TotalNodeTraces(ArrCount, CurRep)
                Else
                    .Cells(ArrCount + 1, CurRep - RepThresh).value = TotalNodeTraces(ArrCount, CurRep)
 - TotalNodeTraces(ArrCount - 1, CurRep)
                End If
            End If
        Next ArrCount

        If CurRep = RepThresh + RepsPerSheet And Number_Reps > RepsPerSheet Then
            CurSheetName = "Reps" & Format(RepThresh + 1, "####0") & "to" & Format(CurRep, "####0")
            .Name = CurSheetName
            RepThresh = CurRep
```

```
            'CurSheetNum = ActiveSheet.Number
            Sheets.Add
            '.Move After:=Sheets(CurSheetNum)
        ElseIf CurRep = Number_Reps Then
            CurSheetName = "Reps" & Format(RepThresh + 1, "####0") & "to" & Format(CurRep, "####0")
            .Name = CurSheetName
        End If

    End With


Next CurRep

NewFileNameString = ModelDescString & "_" & ThisTimeString & ".xls"
Range("A1").Select
ChDir _
    CurrDirectoryString
ActiveWorkbook.SaveAs Filename:= _
    CurrDirectoryString & "\" & NewFileNameString _
    , FileFormat:=xlNormal, Password:="", WriteResPassword:="", _
    ReadOnlyRecommended:=False, CreateBackup:=False



End Sub
```