TreeAge Software, Inc. presents:

TreeAge Pro
2-Day Healthcare Training
Day 2

Using TreeAge Pro for
Health Economic Modeling

Course Agenda

Day 2

Training Modules:

6. Markov Models
7. Markov Models & Time Dependence
8. Microsimulation
9. Microsimulation – Decisions & PSA
10. Microsimulation Advanced Techniques
11. Extras

Markov Models

Module Agenda:

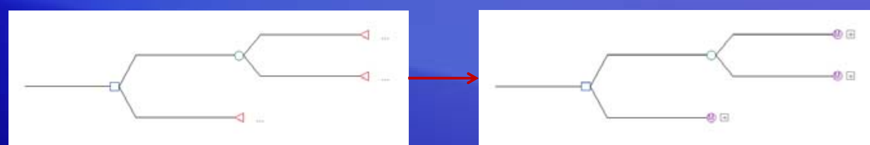
1. **Introduce Markov Models**
2. **Build Markov Model**
3. **Markov Cohort Analysis**
4. **Markov Extras**
5. **Markov Modeling Exercise**

Markov Models

- Markov Models:
  - Also called state transition models
  - Used within TreeAge Pro to track patients over a series of cycles, like …
    - Chronic, progressive diseases
    - Cycles of screening/treatment
- Without a Markov model, a tree would need to have branches to represent each transition at each interval
  - Very large and inefficient

Markov Models

- Markov Models:
  - Consist of the Markov node and everything to the right
  - Evaluate to a single cost and effectiveness measure
    - Can replace any terminal node in a decision tree



  - Markov node *cannot* be placed within another Markov subtree

Markov Models

- Elements of Markov models:
  - Cohort
    - Markov models follow a hypothetical cohort which is split among health states and transitions
    - Usually a hypothetical cohort of 1 to measure expected value for a single person
    - Cohort is homogeneous
  - Cycles
    - Model is run for a time period divided into any number of cycles of fixed length (e.g., 20 1-year cycles)
    - Cycle length is implied (must use probabilities and rewards consistent with cycle length
    - Use built-in _**stage** keyword as reference to the cycle count
      - _stage = 0 during first cycle

Markov Models

- Elements of Markov models (continued):
  - Markov node
    - Starts the Markov model
    - Termination condition indicates when to stop processing
      - Evaluated *before* each cycle
      - Frequently a function of _stage (_stage = 20 runs for 20 cycles)
      - Can run until entire cohort is dead
        - Be careful if prob of death never reaches 100% (will run forever)
      - Multiple conditions? "&" = AND "|" = OR
  - Health States
    - Direct branches from Markov node
    - Track the changing distribution of the cohort among a number of mutually exclusive states
    - Initial probabilities divide the cohort among the health states before the first cycle
    - State rewards (described later)

Markov Models

- Elements of Markov models (continued):
  - Transition Subtrees
    - Tracks what can happen within a cycle
      - Events (e.g., surgery, screening test, adverse event, etc.)
      - Separate transition subtree for each Markov state
    - Transition probabilities and logic drive the cohort through the transition subtree
    - Transition to different states to start the next cycle or stay in the same state
      - Changes the division of the cohort among health states by cycle
      - Implemented through terminal nodes with jump states
    - Absorbing states (e.g., Dead) do not have a transition subtree or jump states

Markov Models

- Elements of Markov models (continued):
  - Rewards
    - Accumulate values (e.g., cost, utility/effectiveness, other)
      - Similar to payoffs at non-Markov terminal nodes
      - Calculation method defines which rewards to accumulate
        - Cost, effectiveness, both, other
      - Can be displayed or hidden within the tree via Tree Preferences
    - State rewards:
      - Accumulated by % of cohort starting a cycle in a health state
        - Example: cost of treating someone with a chronic disease
    - Transition rewards:
      - Accumulated by % of cohort that starts a cycle in a health state AND passes through a specific transition
        - Example:
          - Cost of surgery applied at specific node in transition subtree
          - Not everyone starting in that health state will reach that surgery node

Markov Models

- Elements of Markov models (continued):

State Rewards – 3 Elements
(let's assume 100 cycles or stages)

"Init"	Cost and utility for first model cycle * (not first cycle after transitioning to state) Assigned based on **starting** state membership	**_stage = 0**
"Incr"	Cost and utility for second and all subsequent cycles in process (might be same as "Init.")	**_stage = 1 to 99** .
Final	NOT the last cycle, but *after*; used for atypical rewards¥ based on **ending** state membership	*_stage = 100?? after termination*

* Prior costs can be accounted for in init. reward
¥ For example, counting "deaths"

Markov Models

- Markov model in TreeAge Pro:
  - Example07a-MarkovSimple.trex

Markov state node

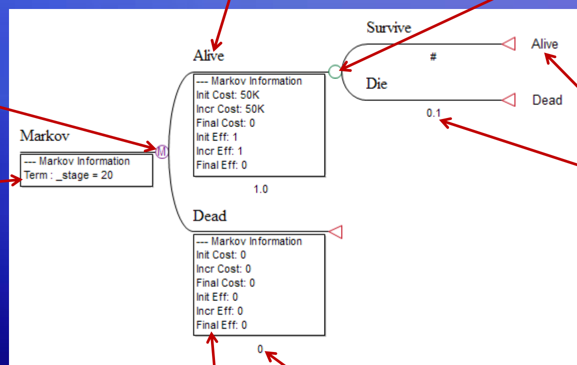
Transition subtree starts here

Markov node

Termination condition

Jump state (for next cycle)

Transition probability



Markov state rewards

Initial probability

Build Markov Model

- We will now build the very simple Markov model from the previous slide
  - Markov node:
    - 1-year cycles (implied)
    - Terminate model after 20 years
  - Markov states:
    - Alive & Dead
    - Entire cohort starts in Alive state (initial probabilities)
    - For each Alive cycle, accumulate…
      - State rewards
      - Effectiveness of 1 LY
      - Cost of $50K
  - Transition subtree:
    - At each annual cycle, there is a 10% chance of death

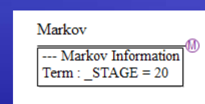
Build Markov Model

Information:

- Markov node:
  - 1-year cycles (implied)
  - Terminate model after 20 years

Instructions:

1. Create new model from toolbar icon (blank tree).
2. Drag Markov node from palette to Tree Diagram Editor.
3. Enter node label text.
4. Open the Markov Info View.
5. Edit the default termination condition.



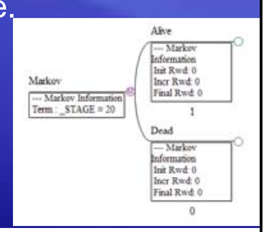
Build Markov Model

Information:

- Markov states:
  - Alive & Dead
  - Entire cohort starts in Alive state (initial probabilities)

Instructions:

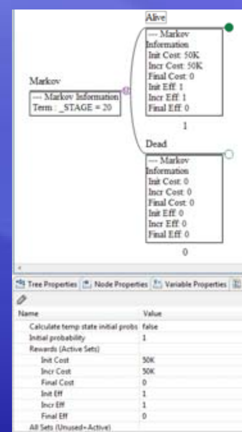
1. Double-click on the Markov node to add two branches (Markov states).
2. Enter node label text for each Markov state.
3. Enter the initial probability beneath each state.



Build Markov Model

Information:

- Markov states:
  - State rewards for Alive state
    - Effectiveness of 1 LY (init and incr)
    - Cost of $50K (init and incr)



Instructions:

1. Edit the Tree Preferences to set the Calculation Method to Cost-Effectiveness and set numeric formatting.
2. Select the Alive state node.
3. Open the Markov Info View.
4. Enter the State Rewards values… initial and incremental rewards.

Build Markov Model

Information:

- Transition subtree:
  - At each annual cycle, there is a 10% chance of death

Instructions:

1. Double-click on the Alive node to add two branches in the transition subtree.
2. Enter the node label text for each branch.
3. Enter the probability for the Live (#) and Die (0.1) branches.

Build Markov Model

"Information":

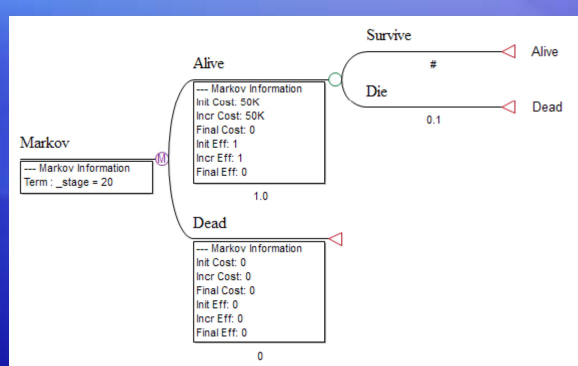
- Need to terminate that transition subtrees

Instructions:

1. Right-click on the Live and change the node type to Terminal.
2. Select the jump state Alive when prompted.
3. Repeat for the Die branch and select the jump state Dead.
4. Repeat for the Dead state – no jump state needed for absorbing state.

Build Markov Model

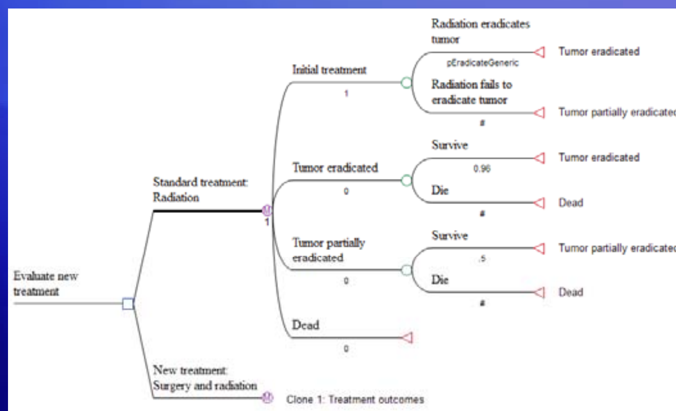
- Model is now complete



Example07a-MarkovSimple.trex

Build Markov Model

- Imagine if we couldn't estimate life expectancy in our Surgery & Radiation model
  - Use Markov model to estimate



Example07a-MarkovPriorTree.trex

Markov Cohort Analysis

- Markov models calculated in two ways:
  - Monte Carlo, patient-level simulation (Microsimulation)…
    - Run patient randomly through the model
    - Accumulate cost and effectiveness for each patient
    - Repeat for many random patients, get means, variances
  - Markov Cohort Analysis
    - Expected value calculation
    - Usually the preferred method
    - No individuals, just a shifting probability distribution among health states (hypothetical "cohort")
- We will focus on Markov Cohort Analysis in this section

Markov Cohort Analysis

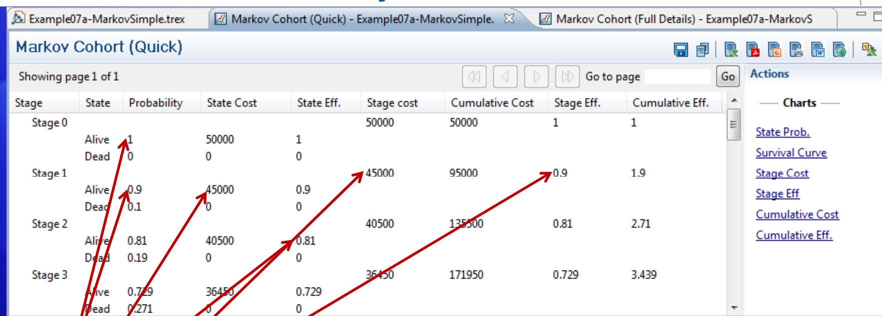
- Markov Cohort Analysis details:
  - For each cycle…
    - For each state…
      - Determine the % of the cohort in the state
        - StateProb
      - Create a reward product from the StateProb and the init or incr reward entered for that state (RwdEntry)
        - StateRwd = StateProb * RwdEntry
    - Sum the reward products for all states
      - $\sum_{states}$ (StateRwd)
  - Sum the state rewards for all cycles
    - $\sum_{cycles}$ ($\sum_{states}$ (StateRwd))
    - Generates the Expected Value
  - Perform for each active payoff

Markov Cohort Analysis

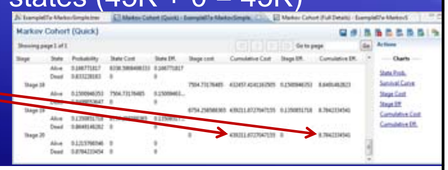
Instructions:

1. Select the Markov node.
2. Choose Analysis > Markov Cohort > Markov Cohort (Quick)

Markov Cohort Analysis



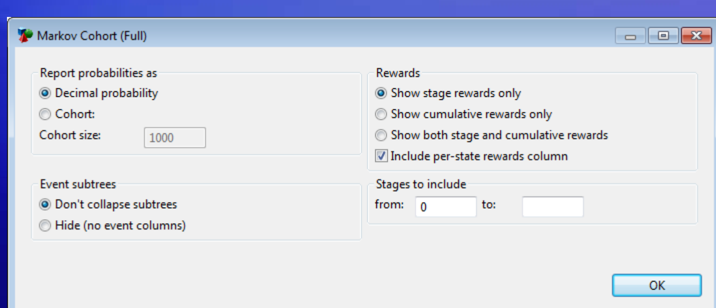
- Markov Cohort (Quick) output…
  - StateProb for each state
  - Reward product for each state/cycle (50K * 0.9 = 45K)
  - Sum of reward products for all states (45K + 0 = 45K)
  - Total EV (all states, all cycles)



Markov Cohort Analysis

Instructions:

1. Select the Markov node.
2. Choose Analysis > Markov Cohort > Markov Cohort (Full)
3. Enter options (if desired) to change data view



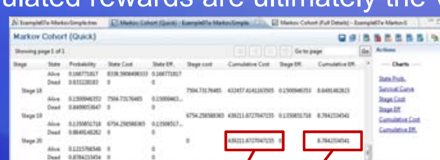
Markov Cohort Analysis

Stage 0				50000	1
Alive	1	50000	1		
Survive	0.9				
jump to: Alive	0.9				
Die	0.1				
Dead	0	0	0		
Stage 1				45000	0.9
Alive	0.9	45000	0.9		
Dead	0.1	0	0		
Stage 2				40500	0.81
Alive	0.81	40500	0.81		
Dead	0.19	0	0		
Stage 3				36450	0.729
Alive	0.729	36450	0.729		
Dead	0.271	0	0		
Stage 4				32805	0.6561
Alive	0.6561	32805	0.6561		
Dead	0.3439	0	0		
Stage 5				29524.5	0.59049
Alive	0.59049	29524.5	0.59049		
Dead	0.40951	0	0		

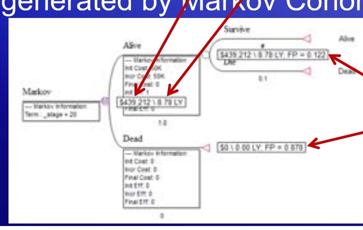
- Markov Cohort (Full) output…
  - Provides additional information on transitions
    - Can be helpful for debugging

Markov Cohort Analysis

- Markov Cohort Analysis:
  - Provides details by cycle and state, but the total accumulated rewards are ultimately the values we need



  - Reflects the overall Expected Value for the Markov model
  - Roll back and other EV analyses use the overall Expected value (generated by Markov Cohort Analysis in background)

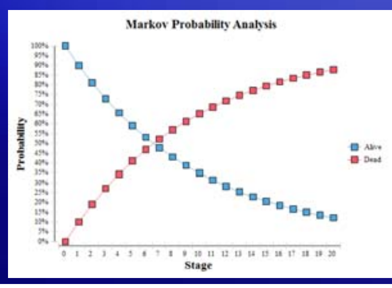


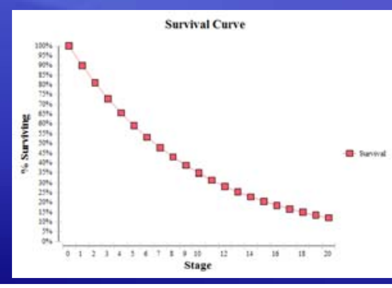
That allows Markov models at endpoints of decision trees

Final StateProb values

Markov Cohort Analysis

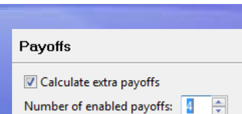
- Markov Cohort Analysis chart options:
  - State Prob – shows breakdown among states by cycle
  - Survival curve – shows percentage of cohort that is alive by cycle (choose dead states when prompted)
  - Stage Cost, Stage Effect – shows rewards by cycle
  - Cumulative Cost, Cumulative Eff. – shows accumulation of rewards by cycle



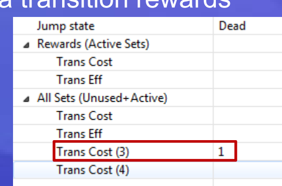


Markov Model Extras

- Using extra payoffs:
  - Tree preferences:



    - Set to calculate extra payoffs beyond the active cost and effectiveness payoffs
    - Can use extra payoffs to count events via transition rewards
      - Reward of 1 will show the percentage of cohort that passes through that transition
        (unless someone could pass through the transition twice)



    - Reflected in Markov Cohort Analysis output
    - Also included in Monte Carlo simulation output
    - Roll back only shows active payoffs
      - Terminal columns can show others

Markov Model Extras

- Half-cycle correction:
  - Markov state rewards provides full cycle reward to cohort that starts cycle in a state
  - Transitions occur at end of cycle
  - Overestimates rewards (e.g., life expectancy)
  - Transitions at mid-point of cycle would be closer approximation to proper reward/survival

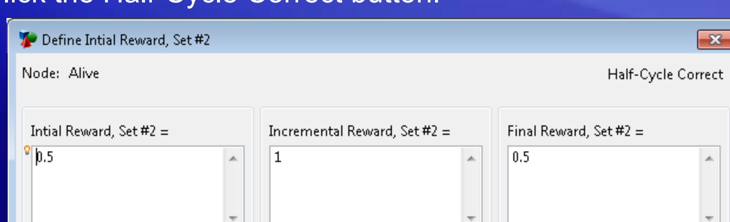
Dies in Cycle…	Eff. Without Corr.	Eff. With Corr.
1	1	0.5
2	2	1.5
3	3	2.5
never	3	3

  - Apply consistently to all reward sets

Markov Model Extras

- Half-cycle correction:
  - Implementation:
    - Apply half reward in initial reward
    - Apply full reward in incremental reward
    - Apply “missing” half reward in final reward

1. Select reward set in Markov Info View.
2. Click pencil icon to open the Reward Set Dialog.
3. Click the Half-Cycle Correct button.



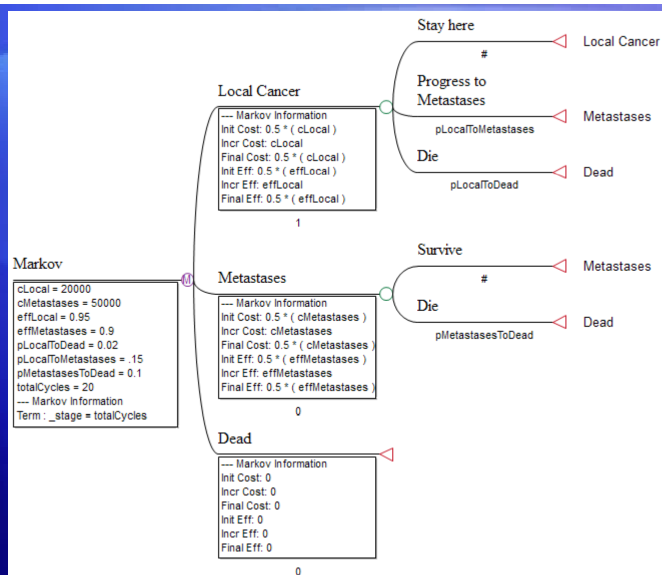
Markov Modeling Exercise

- Exercise: Cancer Progression Model
  - Three states: Local Cancer, Metastases, Dead
  - Local Cancer:
    - Annual mortality = 2%
    - Annual progression to Metastases = 15%
    - Annual cost = $20K
    - Annual effectiveness = 0.95 QALY
  - Metastases state:
    - Annual mortality = 10%
    - Annual cost = $50K
    - Annual effectiveness = 0.90 QALY
  - Dead
    - No cost or effectiveness

Markov Modeling Exercise

- Exercise: Cancer Progression Model
  - 20 one-year cycles
  - Entire cohort starts in Local Cancer state
  - Create variables for all numeric quantities including probabilities and rewards
  - Perform half-cycle correction
  - Use payoff 3 to count progressions to Metastases via a transition reward
  - Use payoff 4 to count deaths from Local Cancer
  - Use payoff 5 to count deaths from Metastases

Markov Modeling Exercise



Example08-MarkovCancer.trex

Markov Models & Time Dependence

Module Agenda:

1. **Time Dependence & Tables**
2. **Time-in-State & Tunnels**

Markov Models & Time Dependence

- In Module 6, the Markov model transition probabilities and rewards were fixed
- Some probabilities/rewards may be…
  - Time-dependent   y = f (_stage)
  - Age-dependent   y = f (_stage + startAge)
  - Time-in-state dependent (next section)

Markov Models & Time Dependence

- Time-dependent transition probabilities
  - If only 2 or 3 possible values, use If( ) or Choose( )…

```
Prob/Rwd = IF(_stage<10; Val_1; Val_2)
Prob/Rwd = Choose(whichVal; Val_1; Val_2; Val_3)
```

  - Otherwise, use tables

```
Prob/Rwd = TableName(index)
```

- Tables allow you to enter a list of values that can be retrieved by an index
  - Retrieve by _stage to use different table value for each cycle

Markov Models & Time Dependence

- Tables in TreeAge Pro:
  - Table properties control behavior
    - Lookup method for missing index values
  - Table data organized by rows & columns
  - Index column is required
  - Multiple value columns allowed
    - Can rename value columns, but not “Index” column
  - Part of the model *.trex file
    - Not default for TreeAge Pro 2009
  - Export/import:
    - Export to global table files
    - Import into another model
  - Attach to external data source via ODBC

Markov Models & Time Dependence

- Tables in TreeAge Pro:
  - Values are retrieved by index and value column

```
TableName[index; valueColumn]
TableName[20; 2] = 2000
TableName[30] = 300
```

Index	Value	Value 2
10	100	1000
20	200	2000
30	300	3000
40	400	4000
50	500	5000

- If value column not provided, first column is assumed
- Note the square brackets in table lookups
- Interpolation:

```
TableName[32] = 320
TableName[20; 1.5] = 1100
```

Markov Models & Time Dependence

- We will now incorporate tables and time-dependent probabilities into our Cancer model
- Changes/assumptions:
  - Add transition before disease-related progression and/or death to account for background mortality
  - Use new mortality tables for probabilities of death from background mortality
  - Assume cohort starts at age 50

Markov Models & Time Dependence

Information:

- Add transition before disease-related progression and/or death to account for background mortality

Instructions:

1. Open Example08-MarkovCancer.tre and save as new document.
2. Hide Variables and Markov info via Tree Preferences (focus on structure).
3. Right-click on Local Cancer node and insert node to the right. Label the node.
4. Add a terminal node beneath the new node.
5. Select the jump state “Dead” and label the node.
6. Repeat steps 3-5 for the Metastases state.

Markov Models & Time Dependence

Information:

- Use new mortality tables for probabilities of death from background mortality
  - First, create the table

Instructions:

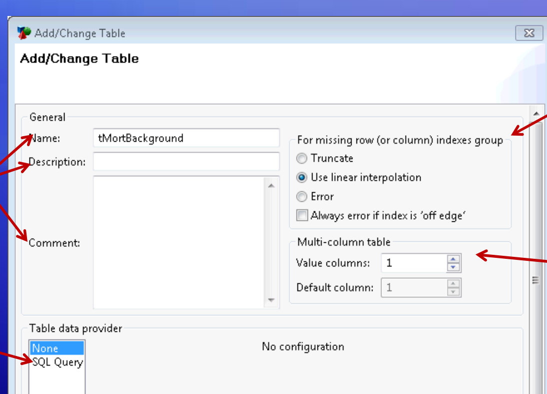
1. Open the Tables View.
2. Click the “+” icon to create a new table.
3. Enter the table name “tMortBackground”.

Markov Models & Time Dependence

- Add/Change Table Dialog:

Name and descriptors

External data source



Lookup method

Column defaults

Markov Models & Time Dependence

Information:

- Use new mortality tables for probabilities of death from background mortality
  - Enter the table data

Instructions:

1. Select the new table in the Tables View.
2. Click the “+” icon in the Table Rows section of the Tables View.
3. Enter the first row of data:
   - Index: 0
   - Value: 0.007

Markov Models & Time Dependence

- We will use the following mortality table data
  - Note that not all ages are in the table
  - Interpolation will handle the gaps
  - tMortBackground[1] = 0.000315
  - tMortBackground[5] = 0.00017
  - tMortBackground[2] …
    = (3/4) * 0.000315 + (1/4) * 0.00017
    = 0.00027875
  - Can use Evaluator View to test

Index	Value
0	0.007
1	0.000315
5	0.00017
15	0.000815
25	0.001036
35	0.002016
45	0.004332
55	0.009409
65	0.02255
75	0.054631
85	0.145933
95	0.25
120	1

Markov Models & Time Dependence

Information:

- Use new mortality tables for probabilities of death from background mortality
  - Enter the table data

Instructions:

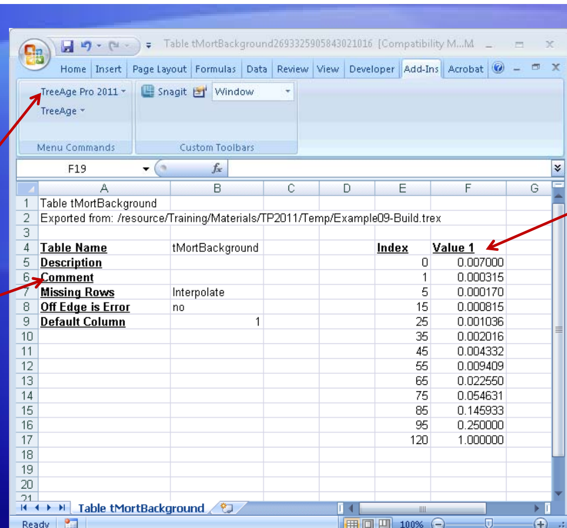
1. Select the new table in the Tables View.
2. Click the “To Excel” icon.
3. Edit the table data in Excel.
4. Excel 2007: Choose Add-ins > TreeAge Pro 2011 > Add or Update Table from Excel.
5. Excel 2003: Choose TreeAge Pro 2011 > Add or Update Table from Excel.

Markov Models & Time Dependence

Send back to TreeAge Pro

Edit properties

Edit data



	A	B	C	D	E	F	G
1	Table tMortBackground						
2	Exported from: /resource/Training/Materials/TP2011/Temp/Example09-Build.trex						
3							
4	**Table Name**	tMortBackground			**Index**	**Value 1**	
5	**Description**				0	0.007000	
6	**Comment**				1	0.000315	
7	**Missing Rows**	Interpolate			5	0.000170	
8	**Off Edge is Error**	no			15	0.000815	
9	**Default Column**	1			25	0.001036	
10					35	0.002016	
11					45	0.004332	
12					55	0.009409	
13					65	0.022550	
14					75	0.054631	
15					85	0.145933	
16					95	0.250000	
17					120	1.000000	
18							
19							
20							

- Look for “To Excel” button in Variables View, Distributions View, etc.
  - Requires Excel Module, TreeAge Pro Suite

Markov Models & Time Dependence

Information:

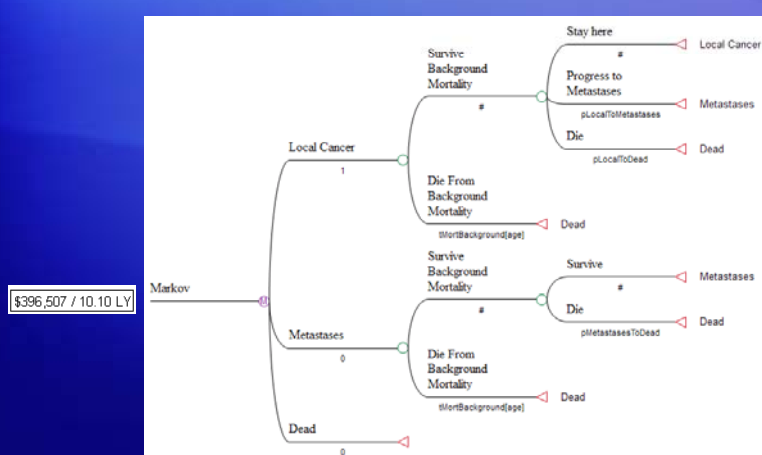
- Use new mortality tables for probabilities of death from background mortality
    - Incorporate the table into the model

Instructions:

1. Define two variables at the root node.
    - startAge = 50
    - age = startAge + _stage
2. Enter the probability of death from background mortality.
    - tMortBackground[age]
    - Use "#" for complement (survival).
3. Repeat step 2 for other background mortality node.

Markov Models & Time Dependence

- Age is a function of _stage
- Probability tMortBackground[age] will vary by cycle



Example09-MarkovCancerTime.trex

Markov Models & Time Dependence

- Discounting:
  - Standard practice for costs, life expectancy/QALYs in multi-year models
  - Apply consistently to all reward sets
  - Discount(value; rate; time) function:
    - value = base cost (or utility) for one cycle
    - rate = discount rate per period (usually annual rate)
    - time = number of periods to discount by (usually _stage
- Different cycle length:
  - Rewards:
    - Multiply/divide by conversion factor
  - Probabilities:
    - Cannot just multiply/divide probability
    - Annual to monthly: ProbToProb(annualProb; 1/12)

Markov Models & Time Dependence

- In the previous section, we looked at factors that depend on time
  - Time-dependent y = f(_stage)
- Now we will look at factors that depend on time-in-state
  - How long a patient has been in a certain state can affect that patient's transitions, etc.
  - For example, the likelihood of metastases may depend on how long the patient has had local cancer
  - Time-in-state dependent y = f(_tunnel)

Markov Models & Time Dependence

- This introduces the concept of a **tunnel**
- A tunnel is a like fixed sequence of temporary states all represented by a single Markov state
  - temporary state 1 (entry point)
  - temporary state 2 (next cycle)
  - …
  - temporary state N (N is max # of tunnels for state)
  - temporary state N
- Within the Markov state …
  - Transition probabilities, rewards, etc. can differ based on the use of the _**tunnel** counter

Markov Models & Time Dependence

- Tunnel implementation in TreeAge Pro:
  - Temporary states are collapsed into one Markov state in model with its rewards and transition subtree
  - Internally the temporary state position is tracked using the time-in-state counter: _tunnel
  - When patient starts first cycle in state, _tunnel = 1
    - Reset to 1 if patient leaves state and returns
    - Use microsimulation/trackers if you do not want reset
  - If the patient stays in the same state, the _tunnel increments by 1 each cycle
  - Probabilities, rewards, etc. can be functions of _tunnel
  - Reports (e.g., Markov Cohort Analysis) can expand into separate columns for each temporary state

Markov Models & Time Dependence

- Why use tunnels?
  - Progressive disease (like cancer):
    - Length of time a patient will spend in a particular disease stage (e.g., localized tumor) is uncertain
    - Probability of transitioning to next disease stage (e.g., metastases) may increase or decrease over time
      - Not age-dependent, but time-in-state dependent
  - Infectious disease:
    - Time since infection may affect prognosis

Markov Models & Time Dependence

- We will now incorporate time-in-state dependent probabilities into our model
- Progression from Local Cancer to Metastases

Index	Probability
1	0.05
2	0.1
3	0.15
4	0.2
5	0.25
6	0.3
7	0.35
8	0.4
9	0.45
10	0.5

- Progression from Metastases to Death

Index	Probability
1	.5
2	1

Markov Models & Time Dependence

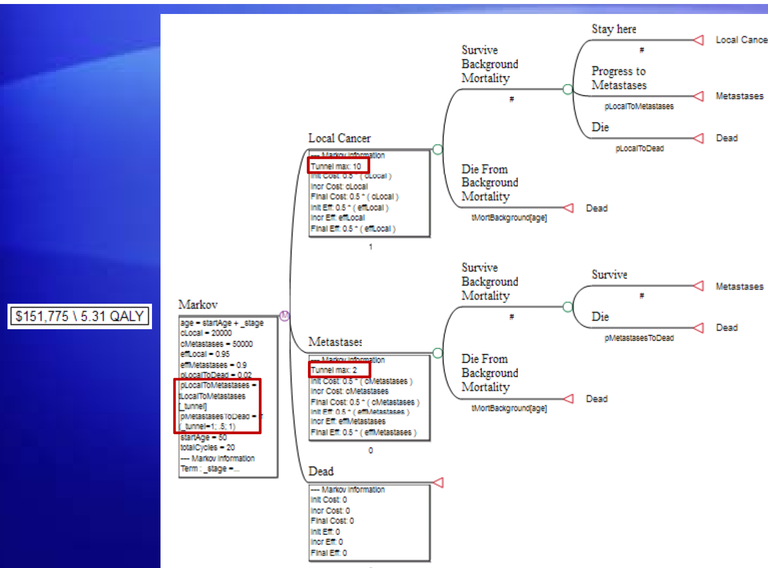
Information:

- Table data for transitions

Instructions:

1. Open Example10-MarkovCancerTime.pkg and save to new file.
2. Select the Local Cancer state.
3. In the Markov Info View, change Tunnel max to 10.
4. Repeat steps 1-2 to set Metastases state Tunnel max to 2.
5. Create table tLocalToMetastases and enter data.
6. Select root node.
7. Change variable definitions in Variable Definitions View.
    - pLocalToMetastases = tLocalToMetastases[_tunnel]
    - pMetastasesToDead = if(_tunnel=1; 0.5; 1)

Markov Models & Time Dependence



Example10-MarkovCancerTunnel.trex

Markov Models & Time Dependence

- Total Expected Value calculations still resolve to a single value for cost and for effectiveness
- Markov Cohort Analysis output can show each tunnel state's temporary states separately or combined

Stage 0					10000	10000	0.475	0.475
	Local Cancer	1	10000	0.475				
	Metastases	0	0	0				
	Dead	0	0	0				
Stage 1					20955.03245	30955.03245	0.9221207407	1.3971207407
	Local Cancer	0.923610435	18472.2087	0.8774299132				
	Metastases	0.049656475	2482.82375	0.0446908275				
	Dead	0.02673309	0	0				

Stage 0					10000	10000	0.475	0.475
	Local Cancer	1	10000	0.475				
	Metastases	0	0	0				
	Dead	0	0	0				
	Local Cancer_Tun_2	0	0	0				
	Local Cancer_Tun_3	0	0	0				
	Local Cancer_Tun_4	0	0	0				
	Local Cancer_Tun_5	0	0	0				
	Local Cancer_Tun_6	0	0	0				
	Local Cancer_Tun_7	0	0	0				
	Local Cancer_Tun_8	0	0	0				
	Local Cancer_Tun_9	0	0	0				
	Local Cancer_Tun_10	0	0	0				
	Metastases_Tun_2	0	0	0				
Stage 1					20955.03245	30955.03245	0.9221207407	1.3971207407
	Local Cancer	0	0	0				
	Metastases	0.049656475	2482.82375	0.0446908275				
	Dead	0.02673309	0	0				
	Local Cancer_Tun_2	0.923610435	18472.2087	0.8774299132				

- Also separate/combined graphical output

Microsimulation

Module Agenda:

1. **Microsimulation**
2. **Trackers**
3. **Microsimulation Exercise**
4. **Microsimulation Analysis**

Microsimulation

- Microsimulation:
  - Monte Carlo simulation technique that generates individual patient histories
  - Individual patients (trials) randomly walk through the model and generate individual outcomes
  - Also known as …
    - Random walk
    - 1st-order simulation
    - 1st-order trial

Microsimulation

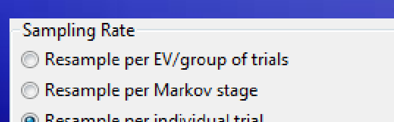
- Microsimulation calculation:
  - Until now, our analysis of Markov models has focused on Expected Value for the entire cohort
  - In contrast, microsimulation generates individual outcomes (cost and effectiveness) for individual patients (trials) based on each random walk
  - By analyzing the aggregate results for a set of trials, we can…
    - Estimate Expected Value (mean)
    - Examine variability among individual outcomes

Microsimulation

- With individual trials, we can now incorporate *individual variability* in the model
  - Individual patient <u>characteristics</u>:
    - Age, gender, ethnicity, etc.
    - Tumor type, tumor size, etc.
    - Can sample from distributions by trial (characteristic, not parameter)
  - Individual patient <u>events</u>:
    - Adverse events
    - Memory of events from prior cycles
    - Use trackers to store values by trial
- Expands modeling capabilities
- Standard cohort analysis cannot account for individual variability because the cohort is homogeneous

Microsimulation

- Sampling distributions by trial:
  - Earlier use of distributions was for *parameter uncertainty*
    - Applied to entire cohort
    - Sampling rate: Once per EV or set of trials
  - Can also use distributions for *individual variability*
    - New sample for each trial at beginning of analysis
    - Use to assign patient characteristics
    - Sampling rate: Once per trial

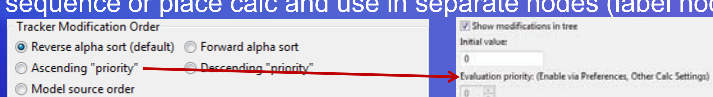


Microsimulation

- Trackers:
  - Trackers store information specific to an individual trial
    - Unlike variables that have a single value
  - Trackers start with an initial value for all trials (usually 0)
  - Tracker values can be retrieved and/or modified for the lifetime of the trial
    - Allows for memory from cycle to cycle
  - Tracker values can be used in expressions of transition probabilities, rewards, etc
    - Just as tunnels can, but trackers are more flexible
  - Avoid using tunnels with microsimulation
    - Trackers can handle all functions for which you would use tunnels
    - Temporary states will slow down the microsimulation

Microsimulation

- Trackers (continued):
  - Definition/modification can reference regular variables, functions, other trackers, etc.
  - Be careful defining a tracker at node where it is used
    - Make sure the trackers are calculated, used in the right sequence or place calc and use in separate nodes (label node)



  - Once transitions depend on trackers, adding more trackers is quick and easy
  - Trackers are only evaluated during Microsimulation
    - Ignored in Expected Value-based analyses

Microsimulation

- Microsimulation – Complications:
  - TreeAge Pro does not report Markov Cohort Analysis output, showing stage-by-stage changes in state probability
    - Advanced: Use Global( ) function to store/report special outputs
  - Cannot run regular Expected Value-based analyses
    - Roll back, N-way sensitivity, etc.
    - Note: run multiple Microsimulation analyses with different variable values
    - Advanced: Possible using Node( ) or Tree( ) function
  - For Microsimulation to provide accurate EV estimates …
    - Need enough repetitions (e.g., tens of thousands or more) for stable mean/std dev values
  - Computationally costly:
    - However, trackers may help keep model small…
    - Not apparent usually until probabilistic sensitivity (2-dim sim)

Microsimulation

- Microsimulation Modeling Exercise:
  - Set individual patient starting age:
    - Generate starting age from uniform distribution (30–50)
  - Track tumor type for each trial:
    - Generate tumor type from a table distribution
      - Less aggressive (70%), prob. of metastases = 0.1
      - More aggressive (30%), prob. of metastases = 0.2
  - If patient survives in Metastases state, there is a 20% chance of having a stroke
    - Probability of death is dependent on the # of strokes
    - Use tracker to count strokes
    - Incorporate into probability

Index	Value1
0	0.1
1	0.15
2	0.25
3	0.4
4	0.6
5	0.85
6	1

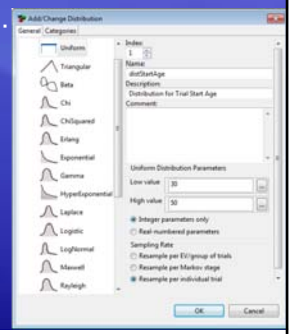
Microsimulation

Information:

- Generate starting age from uniform distribution (30–50)

Instructions:

1. Open Example09-MarkovCancerTime.pkg and save to new file.
2. Open the Distributions View.
3. Click the “+” icon to create a new distribution.
    1. Select type Uniform.
    2. Enter name distStartAge.
    3. Select Integer parameters only.
    4. Enter Low Value & High Value of 30 & 50.
    5. Select Resample per individual trial.



TreeAge Pro Healthcare Training – Module 8 67

Microsimulation

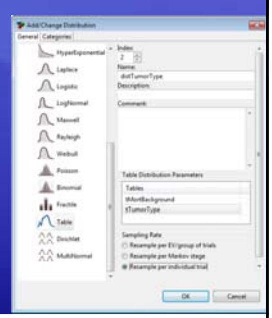
Information:

- Generate tumor type from a table distribution (30%, 70%)

Instructions:

1. Open the Tables View.
2. Click the “+” icon to create a new table.
    1. Enter name tTumorType.
    2. Enter rows 1, 0.7 and 2, 0.3.
3. Open the Distributions View.
4. Click the “+” icon to create a new distribution.
    1. Select type Table.
    2. Enter distTumorType.
    3. Select the table tTumorType
    4. Select Resample per individual trial.

Sum to 100%



TreeAge Pro Healthcare Training – Module 8 68

Microsimulation

Information:

- Generate starting age from uniform distribution (30–50)
- Generate tumor type from a table distribution
  - Less aggressive (70%), prob. of metastases = 0.1
  - More aggressive (30%), prob. of metastases = 0.2

Instructions:

1. Select the root node.
2. Define the variable age as …
   distStartAge + _stage
3. Define the variable pLocalToMetastases as …
   if(distTumorType=1; 0.1; 0.2)
4. Delete the variable startAge.

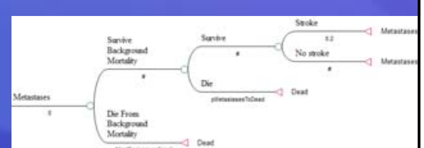
Microsimulation

Information:

- If patient survives in Metastases state, there is a 20% chance of having a stroke
  - Use tracker to count strokes

Instructions:

1. Change the Metastases transition subtree to match this structure.



2. Right-click on the Stroke node and select Define Tracker > New.
   1. Enter the name t_stroke and click OK.
   2. Enter the tracker modification as …
      t_stroke + 1

Microsimulation

Information:

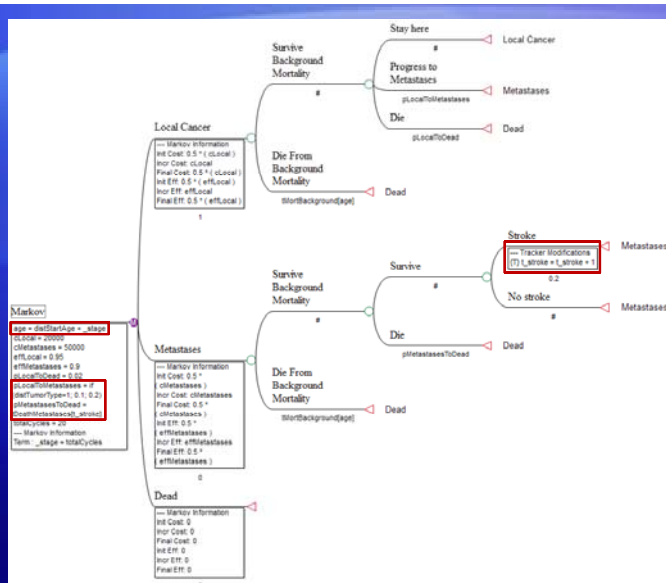
- Probability of death is dependent on the # of strokes

Index	Value1
0	0.1
1	0.15
2	0.25
3	0.4
4	0.6
5	0.85
6	1

Instructions:

1. Open the Tables View.
2. Create table tDeathMetastases with the data above.
3. Select the root node.
4. Define the variable pMetastasesToDead as …
   tDeathMetastases[t_stroke]

Microsimulation



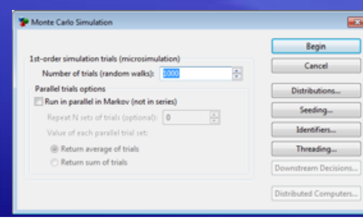
Example11-Microsimulation.trex

Microsimulation

- Our model now …
  - Handles individual characteristics for start age and tumor type
  - Uses a tracker to count strokes and then uses the stroke count in a transition probability expression
- Now we can run the Microsimulation analysis

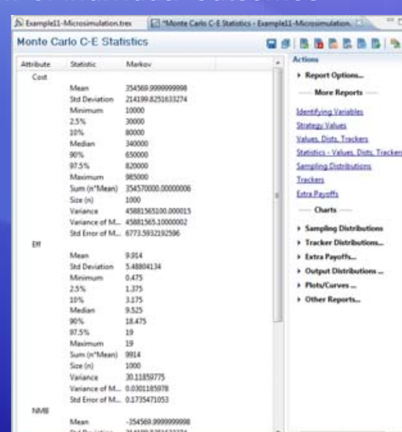
Instructions

1. Select the root node.
2. Choose Analysis > Monte Carlo Simulation > Microsimulation from the menu.
3. Click Begin.



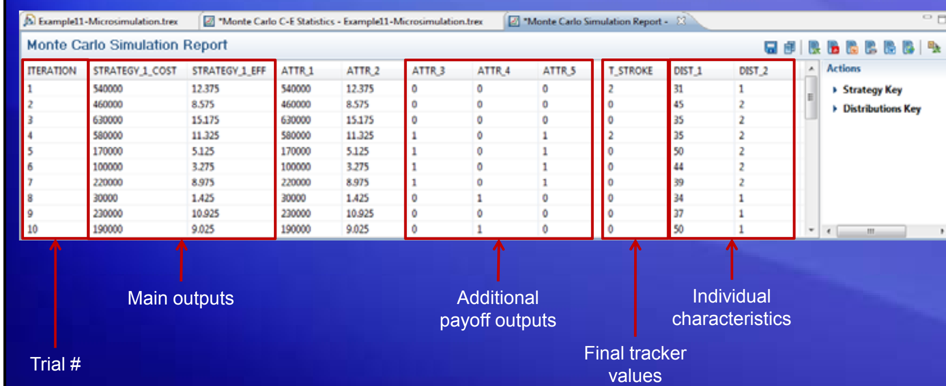
Microsimulation

- Presented with simulation statistics like PSA simulation
  - Statistics are a cumulative view of individual outcomes
  - Mean is estimate of EV
  - Separate calculations for cost and effectiveness
  - If simulation were run at a decision node, separate output for each strategy



Microsimulation

- Values, Dists, Trackers
  - Output for individual outcomes



ITERATION	STRATEGY_1_COST	STRATEGY_1_EFF	ATTR_1	ATTR_2	ATTR_3	ATTR_4	ATTR_5	T_STROKE	DIST_1	DIST_2
1	540000	12.375	540000	12.375	0	0	0	2	31	1
2	460000	8.575	460000	8.575	0	0	0	0	45	2
3	630000	15.175	630000	15.175	0	0	0	0	35	2
4	580000	11.325	580000	11.325	1	0	1	2	35	2
5	170000	5.125	170000	5.125	1	0	1	0	50	2
6	100000	3.275	100000	3.275	1	0	1	0	44	2
7	220000	8.975	220000	8.975	1	0	1	0	39	2
8	30000	1.425	30000	1.425	0	1	0	0	34	1
9	230000	10.925	230000	10.925	0	0	0	0	37	1
10	190000	9.025	190000	9.025	0	1	0	0	50	1

Microsimulation

- Additional outputs:
  - Sampling distributions (input distributions)
  - Output distributions:
    - Cost, effectiveness, NMB, trackers
  - For decision trees:
    - Cost-effectiveness analysis using mean values
  - For 2-dimensional simulations:
    - PSA output described yesterday
  - For both of the above:
    - Distribution of incremental outputs (e.g., ICER)

Microsimulation – Decisions & PSA

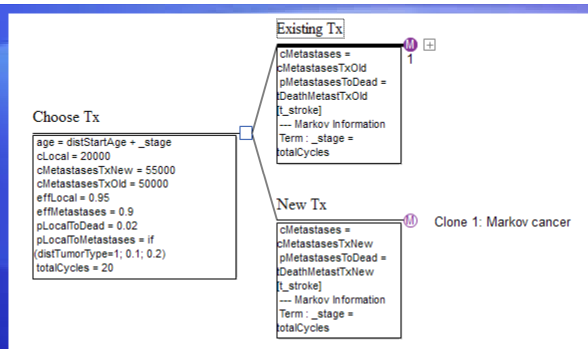
Module Agenda:

1. **Decision Trees & Microsimulation**
2. **Two-Dimensional Simulation**

Microsimulation – Decisions & PSA

- **Usually, Microsimulation is required based on the complexity of the model and individual variability**
- **You still likely have a decision node with alternative treatment options**
- **Model can include both decision tree logic and Microsimulation**
  - **Microsimulation generates statistics (especially mean) for each strategy**
  - **Decision tree compares the strategies**

Microsimulation – Decisions & PSA

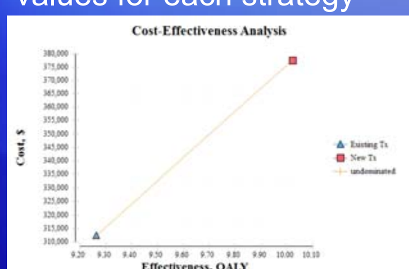


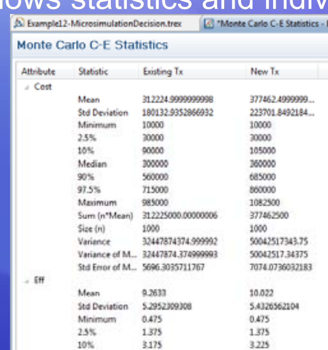
Example12-MicrosimulationDecision.trex

- Note different values for New Tx clone copy
  - Via Variable Definitions
- Run Microsimulation at Decision node (root) to run Microsimulation for each strategy

Microsimulation – Decisions & PSA

- Microsimulation output now shows statistics and individual values for each strategy





Example12-MicrosimulationDecision.trex | *Monte Carlo C-E Statistics

Monte Carlo C-E Statistics

Attribute	Statistic	Existing Tx	New Tx
Cost			
	Mean	312224.9999999998	377462.4999999...
	Std Deviation	180132.9352866932	223701.8492184...
	Minimum	10000	10000
	2.5%	30000	30000
	10%	90000	105000
	Median	300000	360000
	90%	560000	685000
	97.5%	715000	860000
	Maximum	985000	1082500
	Sum (n*Mean)	312225000.00000006	377462500
	Size (n)	1000	1000
	Variance	32447874374.999992	50042517343.75
	Variance of M...	32447874.374999993	50042517.34375
	Std Error of M...	5696.3035711767	7074.0736032183
Eff			
	Mean	9.2633	10.022
	Std Deviation	5.2952309308	5.4326562104
	Minimum	0.475	0.475
	2.5%	1.375	1.375
	10%	3.175	3.225

- Cost-effectiveness graph
  - Uses mean values as estimate of EV
- Do not use PSA-specific outputs
  - ICE scatterplot, Acceptability Curve, Dist of Incrementals
  - Need cohort-level results for PSA outputs
    - Parameter uncertainty

Microsimulation – Decisions & PSA

- Microsimulation is not an EV calculation, so 1-way, 2-way, etc. sensitivity analysis is not directly supported
- However, you can still run PSA on the model via a 2-dimensional simulation
  - Outer loop for parameter uncertainty (samples, 2nd-order)
  - Inner loop for individual variability (trials, 1st-order)
- Can take a long time…
  - Total iterations = samples * trials

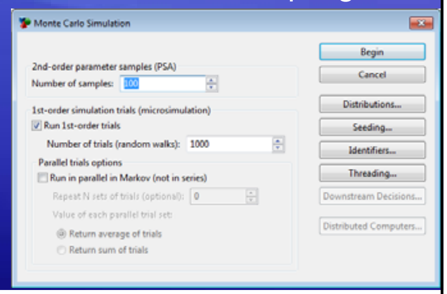
Microsimulation – Decisions & PSA

- How 2-dimensional simulation runs …
  1. Sample parameter uncertainty distributions
     1. Sample individual variability distributions
     2. Run trial
     3. Repeat 1.1 and 1.2 until set of trials is complete
     4. Aggregate into mean values for the trial set
  2. Repeat 1 until set of samples is complete
  3. Aggregate values and present as PSA output
- Results look the same as regular PSA without trial loop
  - Acceptability curve, distribution of incrementals, etc.
  - Lose information on trial-level data/variance (only means)

Microsimulation – Decisions & PSA

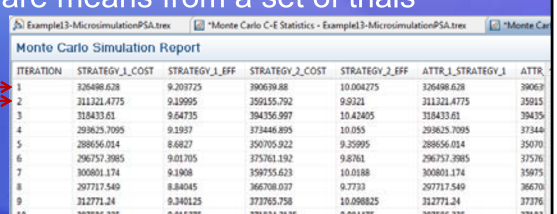
Instructions:

1. Open the Example13-MicrosimulationPSA.trex model.
2. Open Distributions View and check sampling rates.
   1. Distributions 1, 2 are for individual variability.
   2. Distributions 3, 4, 5 are for parameter uncertainty.
3. Select root node.
4. Choose Analysis > Monte Carlo Simulation > Sampling & Trials from the menu.
5. Click Begin.



Microsimulation – Decisions & PSA

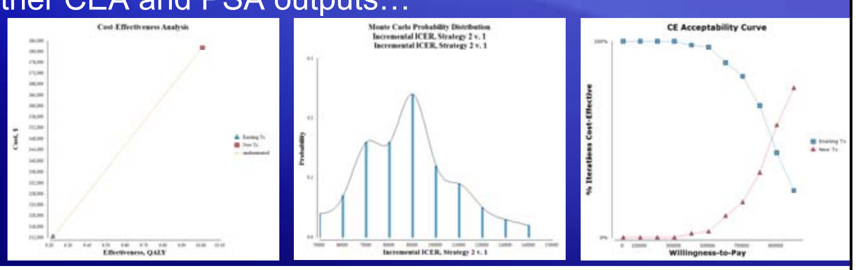
- Results are the same as PSA without trials except that each iteration's values are means from a set of trials rather than EV calcs


Example13-MicrosimulationPSA.trex | *Monte Carlo C-E Statistics - Example13-MicrosimulationPSA.trex | *Monte Car

Monte Carlo Simulation Report

ITERATION	STRATEGY_1_COST	STRATEGY_1_EFF	STRATEGY_2_COST	STRATEGY_2_EFF	ATTR_1_STRATEGY_1	ATTR
1	326498.628	9.203725	390639.88	10.004275	326498.628	39063
2	311321.4775	9.19995	359155.792	9.9321	311321.4775	35915
3	318433.61	9.64735	394356.997	10.42405	318433.61	39435
4	293625.7095	9.1937	373446.895	10.055	293625.7095	37344
5	288656.014	8.6827	350705.922	9.35995	288656.014	35070
6	296757.3985	9.01705	375761.192	9.8761	296757.3985	37576
7	300801.174	9.1908	359755.623	10.0188	300801.174	35975
8	297717.549	8.84045	366708.037	9.7733	297717.549	36670
9	312771.24	9.340125	373765.758	10.098825	312771.24	37376
10	307586.335	9.015275	371824.3125	9.904475	307586.335	37182

- Other CEA and PSA outputs…



Markov/Micro. – Advanced Techniques

Module Agenda:

1. **Discrete Event Simulation**
2. **Parallel Trials**
3. **Bootstrapping with Patient Data**
4. **Dynamic Cohort**
5. **EVPPI Simulation**

- We will just touch on these advanced topics in case you need them in the future

Markov/Micro. – Discrete Event Simulation

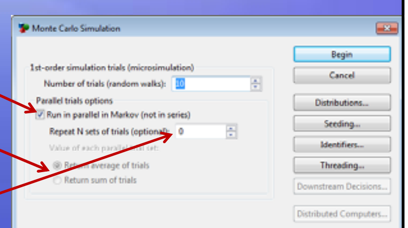
- Microsimulation is DES, but models usually have a fixed cycle length (“time slice” approach)
  - Sometimes “time-to-event” more efficient or natural
  - Abandon _stage counter and fixed cycle length
- Track time using a tracker
  - Increment time as it elapses
    - t_time = t_time + X
  - Time-dependent values are now a function of t_time
    - e.g., prob = Table[t_time]
- Example model: Parallel Trials _CLOCK 1.trex
- Published examples:
  - Barton, et al: BRAM arthritis model
  - LeLay, et al: Depression model

Markov/Micro. – Parallel Trials

- Trials are normally independent, run *in series*
- In some models, an individual trial's transitions might depend on the other trials
  - e.g., infectious disease, organ transplant availability
- Interaction among trials requires running trials *in parallel*
  - _stage 0: trial #1, trial #2… trial #N
  - _stage 1: trial #1, trial #2… trial #N
  - _stage… meet termination condition
- To synchronize trials by time tracker (DES), rather than _stage, use special tracker name: _CLOCK
- Run at Markov node, or decision node with only Markov branches
- Example model: Parallel Trials _CLOCK 1.trex

Markov/Micro. – Parallel Trials

- To run parallel trials…
  - Returns either average or sum of all trial outcomes in set
  - Iterations will show a single line for each set of trials
  - Use sets options to run multiple sets of trials
    - Sometimes necessary to generate stable means



- Published example:
  - "Cost Effectiveness of Combination Antiretroviral therapy for HIV Disease", K. Freedberg, M. Weinstein, S. Goldie, et al. NEJM, Vol 344, No. 11 (2001). www.nejm.org
    - Based on Multicenter AIDS Cohort Study

Markov/Micro. – Bootstrapping

- Use real patient data as input to model
  - Create table with patient data
    - Each row is a patient
    - Each column is a different characteristic
  - Pull data from table for each patient characteristic
    - Draw each patient randomly from the table
      - Via uniform distribution – PatientData[distUniform]
    - Run for each patient in table (possibly more than once)
      - Via _trial keyword – PatientData[_trial]
      - PatientData[Modulo(_trial; tableSize)]

Markov/Micro. – Dynamic Cohort

- Add/subtract from cohort during analysis
  - Works for Markov Cohort Analysis and Microsimulation
  - Examples: infectious disease, population planning, budget analysis
  - Set Tree Preferences/Other Calc Settings to allow non-coherent probabilities (sum <> 100%)
  - Initial probabilities:
    - Number of patients starting in each state
  - Transition probabilities:
    - Can increase/decrease cohort size during any cycle (e.g., births, migration)

Markov/Micro. – Dynamic Cohort

- Often use entry state (not a real health state) to handle the addition of new people into the model
- StateProb( ) function
  - Returns the state probability of one or more states at the start of the current cycle
  - Can be used when probability of infection is dependent on the number (or %) of people in infected state
- Also can have dynamic number of trials with Microsimulation via Parallel Trials
- Example models:
  - Dynamic Population v2008.trex
  - Markov Dynamic Population 2.trex

Markov/Micro. – EVPPI Simulation

- Expected value of partial perfect information (EVPPI)
  - Isolate specific distribution(s) within PSA simulation in outer loop
  - Then sample other distributions in inner loop
    - Aggregated into means
  - Possibly also trials in “most inner” loop
    - Also aggregated into means
  - See isolated impact of specific distribution(s) within the overall PSA simulation
  - 3-dimensional simulations can run slow……..

Extras

Module Agenda:

1. **Testing & Debugging**
2. **Simulation Probabilities**
3. **Simulation Options**
4. **Subgroups**
5. **GlobalN Function**
6. **Interfaces**
7. **Getting Help**

Extras – Testing & Debugging

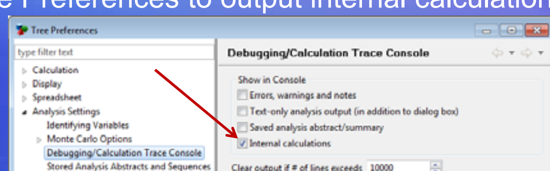
- You may want/need to verify that a model is calculating values as designed
  - Complex formulas, functions, non-root definitions
  - Time-dependent values: tables, functions
  - Markov transitions
  - Assumptions (calibration)
- Temporarily change Markov assumptions …
  - Change starting state using initial probabilities
    - Force entire cohort (or all trials) into a specific state to focus on transitions/calculations in that state
  - Change transition probabilities to force more trials through an unlikely scenario
    - Hard to check calculations if trials almost never hit certain conditions in a model

Extras – Testing & Debugging

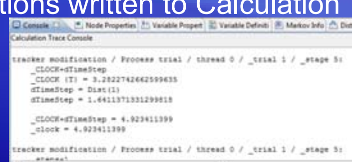
- Sensitivity analysis
  - Use extreme values
  - Look for unexpected changes in effects and costs
- Evaluator View
  - Calculate variable/expression values at selected node
- Output data
  - Add extra trackers for microsimulations to check events in iteration output
- GlobalN function
  - Output data to Global matrices during analysis
  - Dump global matrices at end of analysis
  - Later …

Extras – Testing & Debugging

- Calculation Trace Console
  - Set Tree Preferences to output internal calculations



  - Calculations written to Calculation Trace Console



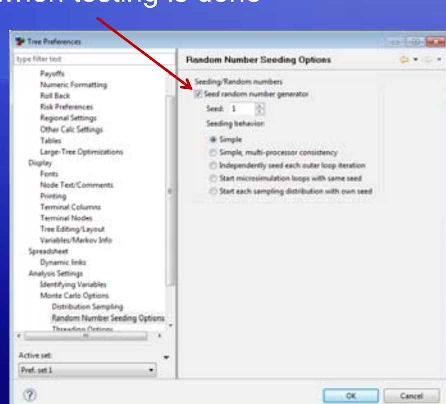
  - Slows down analyses
    - Test Microsimulation with just a few trials

Extras – Simulation Probabilities

- Roll back may run fine, but simulations can still fail
- Probability sampling can generate invalid probabilities
  - Single probability < 0 or > 1
    - Beta distributions bounded by 0 and 1
  - Sum of branch probabilities < 0 or > 1
    - Dirichlet distribution generates any number of coherent probabilities
      - Parameter: List(10; 20; 30; 40)
      - References: Dist(1; 1), Dist(1; 2), Dist(1; 3), Dist(1; 4)

Extras – Seeding Simulations

- Simulations will generate different results every time
- Use seeding to get repeated results
  - Useful for testing, but do not overuse
  - Turn off when testing is done

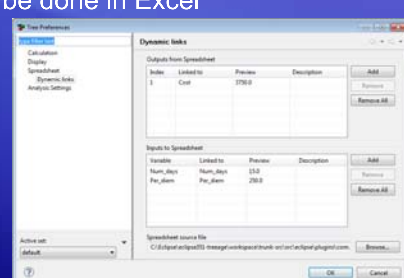


Extras – Highlighted Functions

- GlobalN functions:
  - Store and retrieve data at any time within a tree
  - Among its uses…
    - Facilitates interaction among parallel trials
    - Store Markov transitions in a microsimulation
    - Store tracker at specific point in transition (microsimulation)
    - Output extra data from analyses not provided by TreeAge Pro
  - Store: GlobalN(index; row; column; data)
  - Retrieve: GlobalN(index; row; column)
  - Export to Text: GlobalN(index)
  - Export to Excel:
    Command("EXCEL"; "ExportGlobalMatrixN"; index)
  - Example model: Global Function (simple).trex

Extras – Interfaces

- Bilinks:
  - One direction:
    - Pull data from Excel into model
  - Both directions
    - Send data to specific Excel cells based on location in model
    - Calculate other cells in Excel
    - Pull calculated data back into TreeAge Pro
    - Allows complex calculations to be done in Excel
    - Slows model analysis,
      so use only when required



Extras – Interfaces

- Object Interface:
  - Provides access to TreeAge Pro objects/functions from any program/script/macro
  - Provides access to most of TreeAge Pro's core functions
  - Requires the Excel Module (TreeAge Pro Suite)
  - Java or ActiveX connection
  - Uses
    - Automate tasks that are common, repetitive and/or time-consuming
    - Set a tree's data values from outside TreeAge Pro
    - Automatically run a set of analyses
    - Automatically export analysis results
    - Integrate TreeAge Pro functions into another system or application

Extras – Getting Help

- Context-sensitive help/manual
  - F1 or from Help menu
  - Complete description of most features
- Technical support
  - Included with active license
    - Maintenance must be active for standard/perpetual license
  - support@treeage.com
  - 413-458-0104, then 1 for support
- Online training
  - For more extensive support than what is covered by Technical Support
  - Via GoToMeeting service