Steady-State Power System Security Analysis with PowerWorld Simulator

S9: ATC/PVQV Automation

2001 South First Street
Champaign, Illinois 61820
+1 (217) 384.6330

support@powerworld.com
http://www.powerworld.com

General Automation

- Simulator Auxiliary Files
  - Auxiliary files can be used to automate repetitive processes
  - Various script commands exist that will perform most functions available through GUI
  - Batch process with no looping structure
  - No process control when an error is encountered

General Automation

- SimAuto
  - COM automation server that allows Simulator to be controlled from an external application
  - Use Simulator as a solution engine only
  - Looping structure and process control available that is not available with auxiliary file batch processing
  - SimAuto functions and all Simulator script commands can be used

ATC Automation

- Two different ways of automating a simple ATC study will be demonstrated
  - SimAuto application in Excel
    - *ATC_SimAuto_Example.xls*
  - Master auxiliary file and supporting auxiliary files
    - *ATC_Automation_Example.aux*

ATC Automation

- What do we want to automate?
  - Linear ATC calculations for a number of transfer directions
  - Define a different set of single transmission line contingencies based on area definitions for each transfer direction
  - Define a different set of monitored transmission lines based on area definitions for each transfer direction

ATC Automation

- What do we want to automate?
  - Define different PTDF and OTDF cutoff values for each transfer direction
  - Define different maximum MW transfer cutoff values for each transfer direction
  - Save results to file or load automatically in Excel

ATC Automation

- What do we need for calculations?
  - Base power flow case (PWB file)
  - List of transfer directions and direction specific parameters
  - Auxiliary files for setting options

ATC Automation SimAuto Example

Click to start analysis

Set to *NO* to skip a direction (green indicates direction is processing)

Parameters that vary by transfer direction

Messages indicate if error occurred for base auxiliary files or initial power flow solution

Calculate ATC

Direction	Seller	Seller Type	Buyer	Buyer Type	Calculate	Monitor Areas	Monitor kV	Contingency Areas	CTG kV	Cutoff PTDF %	Cutoff OTDF %	Max Transfer	Processed	Direction Error Message	Error Message
EAST to NI	39	AREA	63	AREA	NO	39	200	39	100	5	5	4000	NO		
EMO to IP	56	AREA	57	AREA	NO	56-63	200	56,57	100	3	3	2000	NO		
TVA to SERC	47	AREA	55	AREA	NO	47-55	100	47-55	100	5	5	6000	NO		
MAPP to WUMS	MAPP	INJECTIONGROUP	WUMS	INJECTIONGROUP	NO	64-68	100	64-68,88	100	3	3	2000	NO		
BUS 33 to 40	33	BUS	40	BUS	YES	39	100	39	100	3	3	6000	NO		

Directions | Input_Files | EAST to NI | EMO to IP | MAPP to WUMS | TVA to SERC | BUS 33 to 40

Inputs

ATC results by direction

Automatically set to *YES* when direction is finished

Messages indicate if error occurred for an individual direction

ATC_SimAuto_Example.xls

ATC Automation SimAuto Example

- Inputs include auxiliary files that will be loaded prior to the analysis to ensure consistent options as a starting point

B9

	A	B
1	Working Directory	...\S09_ATC_PVQV_Automation
2	Base Case	Midwest_Small.pwb
3	Base Case Auxiliary File	Midwest_Base_Case_Options.aux
4	Initial Limit Monitoring Settings Auxiliary File	Midwest_Initial_Limit_Monitoring_Settings.AUX
5	ATC Options Base Auxiliary File	ATC_Options.AUX
6	CTG Options Base Auxiliary File	Midwest_CTG_ATC_Options.AUX
7		

Directions | Input_Files

ATC_SimAuto_Example.xls

ATC_SimAuto_Example.xls

ATC Automation SimAuto Example

- Create looping structure in VBA to process list of directions for which Calculate = YES
- Use SimAuto functions and Simulator script commands to
  - Open base case
  - Load input auxiliary files
  - Set direction specific options
  - Calculate ATC values
  - Present results in Excel worksheets

ATC Automation SimAuto Example

- Key SimAuto Functions
  - OpenCase(FileName)
  - ProcessAuxFile(FileName)
  - RunScriptCommand(Statements)
  - GetParametersMultipleElement(ObjectType, ParamList,FilterName)
  - ChangeParametersSingleElement(ObjectType, ParamList, Values)
  - ChangeParametersMultipleElement(ObjectType, ParamList,Values)

ATC Automation SimAuto Example

- Key Script Commands
  - EnterMode(mode,submode)
    - Need to change modes depending on the script command
  - SolvePowerFlow(SolMethod,"filename1", "filename2", CreateIfNotFound1, CreateIfNotFound2)
    - Make sure that the power flow is solved before continuing with calculations

ATC Automation SimAuto Example

- More Key Script Commands
  - CTGAutoInsert
    - Create single transmission line contingencies based on options set with CTG_AutoInsert_Options
  - ATCDetermine([transactor seller], [transactor buyer], ApplyTransfer)
    - Calculate ATC values for each transfer direction

ATC Automation SimAuto Example

- Key Data Objects
  - AREA, ZONE, OWNER
    - Used with Limit Monitoring Settings and Area/Zone/Owner Filters for use with CTG_AUTOINSERT_OPTIONS
  - FILTER
    - Create a BRANCH filter to limit the number of branches for which contingencies are created for use with CTG_AUTOINSERT_OPTIONS
  - CTG_AUTOINSERT_OPTIONS
    - Set these options before calling the CTGAutoInsert script command

ATC Automation SimAuto Example

- More Key Data Objects
  - ATC_OPTIONS
    - Set PTDF cutoff, OTDF cutoff, Max MW Transfer for each direction
  - TRANSFERLIMITER
    - These are the ATC results
  - Other objects needed in setup auxiliary files for initializing solution options correctly including SIM_SOLUTION_OPTIONS, LIMIT_MONITORING_OPTIONS, LIMITSET, etc.

ATC Automation SimAuto Example

- Updating ATC Options

```
Public Function UpdateATCOptionsViaSimAuto(ThePTDF, TheOTDF,
                        TheMaxTransfer As String) As Boolean
  Dim ParamList, ValueList, output As Variant

  ParamList = Array("ATC_MaxMWLimit", "ATC_IgnorePTDFBelow",
                        "ATC_IgnorePTDFBelow:1")
  ValueList = Array(Val(TheMaxTransfer), Val(ThePTDF),Val(TheOTDF))
  output = SimAuto.ChangeParametersSingleElement("ATC_OPTIONS",
                                                 ParamList, ValueList)
  Call ProcessError(output, "Error updating ATC options",
                     MasterErrorMessageRow, ErrorMessageColumn)
  UpdateATCOptionsViaSimAuto = (output(0) = "")
End Function
```

ATC Automation SimAuto Example

- Calculating ATC and extracting results

```
Dim output, ParamList as Variant
Dim fileline as string
⋮
output = SimAuto.RunScriptCommand("EnterMode(ATC)")
fileline = "ATCDetermine(["  + SellerType + " " + Seller + "],[“ +
                              BuyerType +  " " + Buyer + "])"
output = SimAuto.RunScriptCommand(fileline)
⋮
ParamList = Array("CTGViolMaxLine", "LimViolID", "CTGLabel", _
                "LimViolValue","LinePTDF","LinePTDF:1", _
                "LimViolLimit","LineMW","CTGViolMaxLine:1", _
                "ATCIterativelyFound","CTGSkip","LineMonEle", _
                "LimViolLimit:1")
output = SimAuto.GetParametersMultipleElement("TransferLimiter",
                                              ParamList, "“)
```

ATC Automation
Auxiliary File Example

- A series of auxiliary files can also be used to calculate ATC results for multiple directions in an automated manner
- Use of auxiliary files requires sequential batch processing because there is no looping structure

ATC Automation Auxiliary File Example

- Master Auxiliary File
  - *ATC_Automation_Example.aux*
    - Open base power flow case
    - Load in base auxiliary files
    - Call *ATC_Automation_Example_Part2.aux* if the power flow case solves
      - `SolvePowerFlow(RECTNEWT, "ATC_Automation_Example_Part2.aux",STOP, YES,YES);`

ATC Automation Auxiliary File Example

- Supporting Files
  - *ATC_Automation_Example_Part2.aux*
    - Use this to control which directions to run for a given analysis
    - Use the LoadScript command to reference a particular script section in *ATC_Automation_Example_Part3.aux*
      - `LoadScript("ATC_Automation_Example_Part3.aux", DIRECTION_EAST_TO_NI);`
    - Comment out directions using // to skip a direction

ATC Automation Auxiliary File Example

- Supporting Files
  - *ATC_Automation_Example_Part3.aux*
    - Contains individual script sections for each direction
      - `SCRIPT DIRECTION_EAST_TO_NI`, etc.
    - Use LoadAux command to load the auxiliary files that set the direction specific options
      - `LoadAux("AREA_39_AREA_63_Monitoring.aux",YES);`
    - Use the CTGAutoInsert command to create direction specific single contingencies

ATC Automation Auxiliary File Example

– *ATC_Automation_Example_Part3.aux*

- Use the ATCDetermine command to calculate ATC values for a specific direction
  - `ATCDetermine([AREA 39],[AREA 63]);`
- Use the SaveData command to create output files with the ATC results
  -
```
SaveData("EAST_TO_NI_ATC_Output.aux",AUX,
          TRANSFERLIMITER,[CTGViolMaxLine,
          LimViolID,CTGLabel,LimViolValue,
LinePTDF, LinePTDF:1,LimViolLimit,
LineMW,CTGViolMaxLine:1,
          ATCIterativelyFound,CTGSkip,
          LineMonEle,LimViolLimit:1],[]);
```

ATC Automation Auxiliary File Example

- Supporting Files
  - *Midwest_Base_Case_Options.aux*
    - SIM_SOLUTION_OPTIONS – set the base power flow solution options
    - INJECTIONGROUP – create injection groups
  - *Midwest_Initial_Limit_Monitoring_Settings.aux*
    - LIMIT_MONITORING_OPTIONS, LIMITSET, AREA, ZONE, BUS, BRANCH, INTERFACE – set all of the initial limit monitoring that is not direction specific
  - *ATC_Options.aux*
    - ATC_OPTIONS – set all ATC options

ATC Automation Auxiliary File Example

- Supporting Files
  - *Midwest_CTG_ATC_Options.aux*
    - CTG_OPTIONS – set the contingency options for use with the ATC study
    - Can be used to include contingencies in addition to the single contingencies created for each direction
  - *Direction name _Monitoring.aux*
    - AREA - set the direction specific limit monitoring settings
  - *Direction name _CTGAutoInsertOptions.aux*
    - FILTER - create advanced filter used for setting voltage level when selecting direction specific single branch contingencies
    - AREA, ZONE, OWNER - set the Area/Zone/Owner filters for the areas in which to include single contingencies

ATC Automation Auxiliary File Example

- *Direction name _CTGAutoInsertOptions.aux*
  - CTG_AUTOINSERT_OPTIONS - update the direction specific options for automatically inserting single contingencies
- *Direction name _UpdateATCOptions.aux*
  - ATC_OPTIONS - update the direction specific PTDF cutoff, OTDF cutoff, and Max Transfer level
    -
```
DATA (ATC_OPTIONS,[ATC_MaxMWLimit,
  ATC_IgnorePTDFBelow:1,ATC_IgnorePTDFBelow])
{
  2000 3 3
}
```

PV Automation

- Two ways of automating a simple PV study will be demonstrated
  - SimAuto application in Excel
    - *PV_SimAuto_Example.xls*
  - Master auxiliary file and supporting auxiliary files
    - *PV_Automation_Example.aux*

PV Automation

- What do we want to automate?
  - Multiple PV analyses for different transfer directions
  - Different base case can be defined for each transfer direction
  - Supporting auxiliary files are available containing various option settings for each transfer direction
  - Results are written to file or automatically loaded in Excel

PV Automation

- Case background
  - Small equivalized case of the Eastern Interconnect focused on the Midwest region, *Midwest_Small.pwb*
  - Known voltage stability problem exists (existed when this case was current) for the transfer directions and buses being studied
    - MAPP to WUMS transfer
    - Buses near Arpin
  - Voltage stability results can be used to limit the transfer allowed in other studies by creating a proxy flowgate that has a rating equal to the flow on the flowgate at the critical voltage stability limit
    - Flow on Eau Claire – Arpin 345 kV

PV Automation SimAuto Example

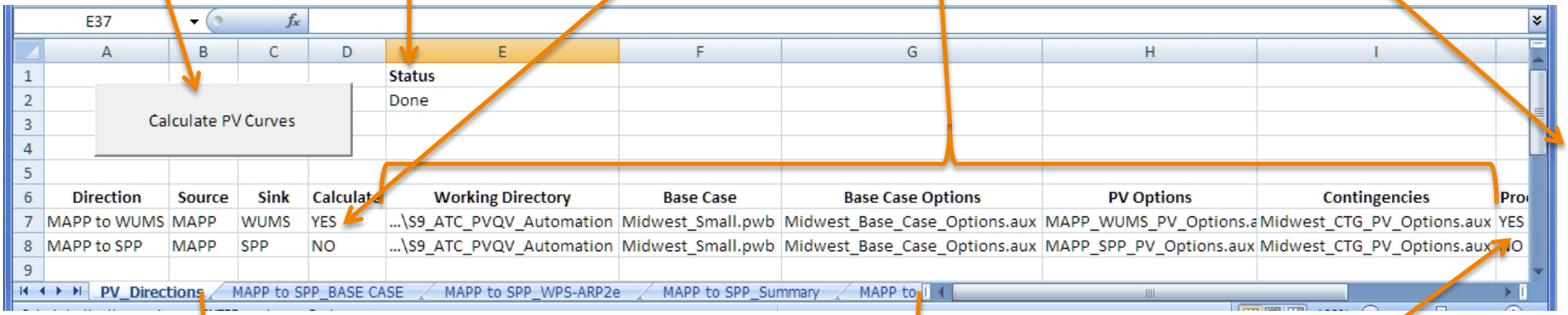
Click to start analysis

Indicate what step the process is in

Set to *NO* to skip a direction

Supporting files

Errors reported in column K



Results - named based on Direction and scenario

Automatically set to *YES* when direction is finished

PV_SimAuto_Example.xls

PV Automation SimAuto Example

- Create looping structure in VBA to process Directions for which Calculate = YES
- Use SimAuto functions and Simulator script commands to
  - Open base case
  - Load auxiliary files
  - Calculate PV curves
  - Present results in Excel worksheets
    - Summary sheet for each direction
    - Tracked results sheet for each scenario

PV Automtion SimAuto Example

- Key SimAuto Functions
  - OpenCase(FileName)
  - ProcessAuxFile(FileName)
  - RunScriptCommand(Statements)
  - GetParametersMultipleElement(ObjectType, ParamList,FilterName)

PV Automation SimAuto Example

- Key Script Commands
  - EnterMode(mode,submode)
    - Need to change modes depending on the script command
  - PVRun([INJECTIONGROUP "sourcename"], [INJECTIONGROUP "sinkname"])
    - Run the PV analysis

PV Automation SimAuto Example

- More Key Script Commands
  - SaveData("filename",filetype,objecttype, [fieldlist], [subdatalist],filter)
    - Create temporary file with results
- Key Data Objects and Supporting Files
  - *Midwest_Base_Case_Options.aux*
    - SIM_SOLUTION_OPTIONS – set base case power flow solution options
    - INJECTIONGROUP – Create injection groups. Source and sink for PV studies must be injection groups.
  - *Midwest_CTG_PV_Options.aux*
    - CTG_OPTIONS – define the options that are specific to solving a contingency
    - CONTINGENCY - create contingencies

PV Automation SimAuto Example

- Key Data Objects and Supporting Files
  - *DirectionName _PV_Options.aux*
    - PVCURVE_OPTIONS – set all options necessary for calculating the PV curves
    - BUS, GEN, INJECTIONGROUP, BRANCH, INTERFACE, TRANSFORMER - define what should be tracked for specific objects at each transfer level
    - AREA - Limit Monitoring Settings
  - PWPVRESULTLISTCONTAINER
    - Contains the summary information for each scenario
    - PWPVRESULTOBJECT subdata object
      - Lists all of the tracked results for each transfer level

PV Automation SimAuto Example

- Running PV Analysis

```
Dim output as variant
Dim fileline, Source, Sink as string
⋮
output = SimAuto.RunScriptCommand("EnterMode(PV)")
fileline = "PVRun("[INJECTIONGROUP " + Chr(34) + Source + _
       Chr(34) + "], [INJECTIONGROUP " + Chr(34) + Sink + _
       Chr(34) + "]) "
output = SimAuto.RunScriptCommand(fileline)
⋮
```

PV Automation SimAuto Example

- Extracting PV results

```
Dim output, ParamList as variant
Dim fileline as string
⋮ The following code creates the temporary file containing the tracking values
fileline = "SaveData(" + Chr(34) + outputfilename + Chr(34) + _
           ", AUX, PWPVRESULTLISTCONTAINER, " + _
           "[PVContingency],[PWPVResultObject])"
output = SimAuto.RunScriptCommand(fileline)

⋮ The following code gets the summary results
ParamList = Array("PVContingency",  "PVQVSecondaryObject",
 "PVResultDescription", "PVResultCritical","PVResultCriticalReason",
 "PVResultMaxShift","PVResultMaxExport","PVResultMaxImport","PVResultNumViol",
 "PVResultWorstVolt","PVResultWorstVoltBus","PVResultCriticalPBusNum",
 "PVResultCriticalPBusName","PVResultCriticalMWMism","PVResultCriticalQBusNum",
 "PVResultCriticalQBusName", "PVResultCriticalMVRMism")
output = SimAuto.GetParametersMultipleElement("PWPVRESULTLISTCONTAINER",
                                              ParamList, "“)
```

PV Automation
Auxiliary File Example

- A series of auxiliary files can also be used to calculate PV results for multiple directions in an automated manner
- Use of auxiliary files requires sequential batch processing because there is no looping structure

PV Automation Auxiliary File Example

- Master Auxiliary File
  - *PV_Automation_Example.aux*
    - Use this to control which directions to run for a given analysis
    - Use the LoadScript command to reference a particular script section in *PV_Automation_Example_Part2.aux*
      - `LoadScript("PV_Automation_Example_Part2.aux", DIRECTION_MAPP_TO_WUMS);`
    - Comment out directions using // to skip a direction

PV Automation Auxiliary File Example

- Supporting Files
  - *PV_Automation_Example_Part2.aux*
    - Contains individual script sections for each direction
      - `SCRIPT DIRECTION_MAPP_TO_WUMS`, etc.
    - Use LoadAux command to load the other supporting auxiliary files that set the direction specific options
      - `LoadAux("MAPP_WUMS_PV_Options.aux",YES);`
    - Use SetData script command to create CSV output file
      - `SetData(PVCurve_Options, [PVCOutFile, PVCSaveToFile], ["MAPP_WUMS_PV_Output.CSV", "YES"]);`

PV Automation Auxiliary File Example

- *PV_Automation_Example_Part2.aux*
  - Use the PVRun script command to run the study
    - `PVRun([INJECTIONGROUP "MAPP"], [INJECTIONGROUP "WUMS"]);`
  - Use the PVWriteResultsAndOptions script command to write out the results to an auxiliary file that can be reloaded into Simulator for later viewing
    - `PVWriteResultsAndOptions("MAPP_WUMS_PV_Results.aux");`

PV Automation Auxiliary File Example

- Supporting Files
  - Files that set base case options, create contingencies, set contingency options, and set PV specific options are the same as used with the PV SimAuto Example as described in the *Key Data Objects and Supporting Files* section.

QV Automation

- Two ways of automating a simple QV study will be demonstrated
  - SimAuto application in Excel
    - *QV_SimAuto_Example.xls*
  - Master auxiliary file and supporting auxiliary files
    - *QV_Automation_Example.aux*

QV Automation

- What do we want to automate?
  - Multiple QV analyses for different base cases
  - Supporting auxiliary files are available containing various option settings for each base case
  - Results are written to file or loaded automatically into Excel

QV Automation

- Case background
  - Power flow cases contain the power transfer at the first critical scenario (reached nose) from the PV runs
    - MAPP to WUMS, 1925 MW
    - MAPP to SPP, 2787.5 MW
  - Use QV analysis to determine how much additional Mvar loading can occur at studied buses
    - Buses originally studied in the PV analysis
    - Buses listed as most violated (Worst V Viol) in the PV analysis

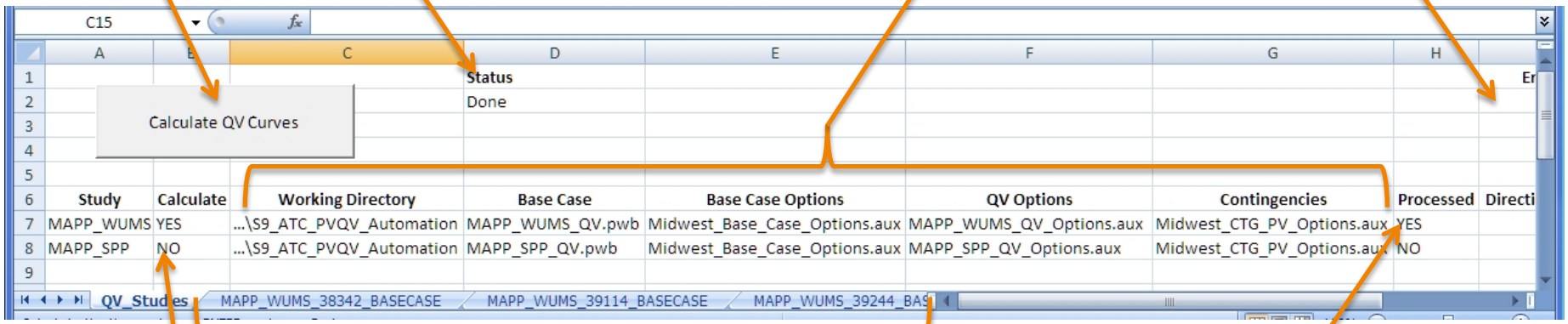
QV Automation SimAuto Example

Click to start analysis

Indicate what step the process is in

Supporting files

Errors reported in column I



Set to *NO* to skip a study

Results - named based on Study and scenario

Automatically set to *YES* when direction is finished

QV_SimAuto_Example.xls

QV_SimAuto_Example.xls

QV Automation SimAuto Example

- Create looping structure in VBA to process Studies for which Calculate = *YES*
- Use SimAuto functions and Simulator script commands to
  - Open base case
  - Load auxiliary files
  - Calculate QV curves
  - Present results in Excel worksheets
    - Summary sheet for each study
    - Tracked results sheet for each scenario

QV Automation SimAuto Example

- Key SimAuto Functions
  - OpenCase(FileName)
  - ProcessAuxFile(FileName)
  - RunScriptCommand(Statements)
  - GetParametersMultipleElement(ObjectType, ParamList,FilterName)

QV Automation SimAuto Example

- Key Script Commands
  - EnterMode(mode,submode)
    - Need to change modes depending on the script command
  - QVRun("filename", InErrorMakeBaseSolvable)
    - Run the QV analysis
    - Save comma-delimited list of results to “filename”

QV Automation SimAuto Example

- Key Data Objects and Supporting Files
  - *Midwest_Base_Case_Options.aux*
    - SIM_SOLUTION_OPTIONS – set base case power flow solution options
  - *Midwest_CTG_PV_Options.aux*
    - CTG_OPTIONS – define options specific to solving a contingency
    - CONTINGENCY - create contingencies

QV Automation SimAuto Example

- Key Data Objects and Supporting Files
  - *StudyName _QV_Options.aux*
    - QVCURVE_OPTIONS – set all options necessary for calculating the QV curves
    - BUS - SELECTED = *YES* for buses for which QV curves should be calculated
    - BUS, GEN, INJECTIONGROUP, BRANCH, INTERFACE, SHUNT, TRANSFORMER – define what should be tracked for specific elements for each study
  - QVCURVE
    - Contains the summary results

QV Automation SimAuto Example

- Running QV Analysis

```
Dim output as variant
Dim outputfilename, fileline, StudyName as string
⋮
output = SimAuto.RunScriptCommand("EnterMode(QV)")
outputfilename = WorkingDirectory + StudyName + "_QVOutput.txt"
fileline = "QVRun(" + Chr(34) + outputfilename + Chr(34) + ", NO)"
output = SimAuto.RunScriptCommand(fileline)
⋮
```

QV Automation SimAuto Example

- Extracting QV Summary Results

```
Dim output, ParamList as variant
⋮
ParamList = Array("BusNum", "BusName", "BusNomVolt", "CaseName",
                  "qv_VQ0", "qv_Q0", "qv_Vmax", "qv_QVmax",
                  "qv_VQmin", "qv_Qmin", "qv_Vmin", "qv_QVmin",
                  "Qinj_Vmax","Qinj_0", "Qinj_min", "Qinj_Vmin")
output = SimAuto.GetParametersMultipleElement("QVCurve",
                                              ParamList, "")
```

QV Automation
Auxiliary File Example

- A series of auxiliary files can also be used to calculate QV results for multiple studies in an automated manner
- Use of auxiliary files requires sequential batch processing because there is no looping structure

QV Automation Auxiliary File Example

- Master Auxiliary File
  - *QV_Automation_Example.aux*
    - Use this to control which studies to run for a given analysis
    - Use the LoadScript command to reference a particular script section in *QV_Automation_Example_Part2.aux*
      - `LoadScript("QV_Automation_Example_Part2.aux", MAPP_TO_WUMS);`
    - Use // to comment out and skip a study

QV Automation Auxiliary File Example

- Supporting Files
  - *QV_Automation_Example_Part2.aux*
    - Contains individual script sections for each study
      - `SCRIPT MAPP_TO_WUMS`, etc.
    - Use LoadAux command to load the other supporting auxiliary files that set the study specific options
      - `LoadAux("MAPP_WUMS_QV_Options.aux",YES);`
      - Load QV options last so that *SELECTED* field is set correctly
    - Use the QVRun script command to run the study
      - `QVRun("MAPP_WUMS_QV_Output.txt", NO);`

QV Automation Auxiliary File Example

- *QV_Automation_Example_Part2.aux*
  - Use the QVWriteResultsAndOptions script command to write out the results to an auxiliary file that can be reloaded into Simulator for later viewing
    - `QVWriteResultsAndOptions("MAPP_WUMS_QV_Results.aux");`
- Files that set base case options, create contingencies, set contingency options, and set QV specific options are the same as used with the QV SimAuto Example as described in the *Key Data Objects and Supporting Files* section.