Tutorial 4: Transmission Losses

This tutorial explains how transmission losses are included in the electricity market model and the impact that including transmission losses has on the results.

Modelling transmission losses

Transmission losses = Power lost as heat

When electrical power flows through a conductor the movement of electrons causes the conductor to heat up. The more power that flows, the more heat. If a conductor gets too hot, then its structure is permanently altered… the maximum flow limit is set to prevent this from happening.

Power that is lost as heat is power that is not transmitted to the bus at the other end of the branch. In the model, this lost power is referred to as branch losses.

Our market model uses the simplified power flow equation to calculate a branch's power flow. The branch loss calculation corresponding to the simplified power flow equation is shown in Equation 9.

$$Losses_{branch} = Resistance_{branch} \times MWFlow^2_{branch}$$

Equation 9: Branch loss calculation for simplified AC power flow

Modelling branch losses

Because the branch loss calculation in Equation 9 includes a quadratic term, it represents a curve (specifically a parabola) and therefore it cannot be directly included in the LP (*Linear* Programming) model used by the simplex algorithm because this model requires its constraints to be linear, i.e., straight lines not curves.

To meet the requirement for linear constraints, the loss curve is modelled as a series of straight lines. To see this, build a two-bus, one-branch model (tap Bus-Bus-Gen-Load-Branch) and solve with the "Include Losses" option set to ON as shown in Figure 42. The result is shown in Figure 43.

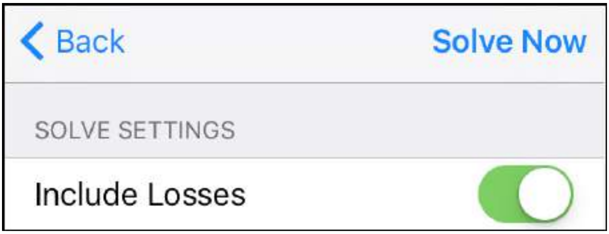

Figure 42: Setting branch losses to ON

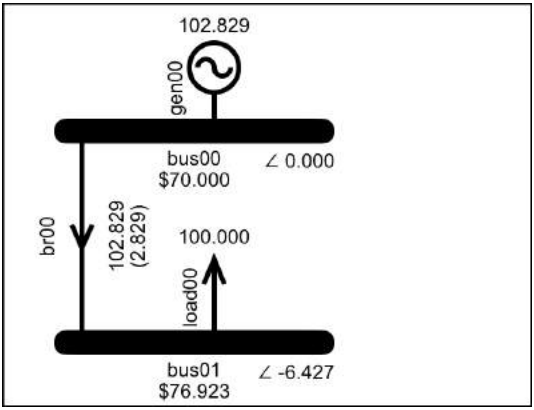

Figure 43: Result that demonstrates branch losses

To view the branch loss parabola, double tap the branch to see its Data Display then tap the "Loss" button on the toolbar, as indicated in Figure 44

Figure 44: Branch Data Display with Loss button indicated

The "Loss" button leads to the "Branch Segments" display in Figure 45, which shows the individual loss segments in table form and also as a plot. The display also shows the settings that were used to calculate the segments.

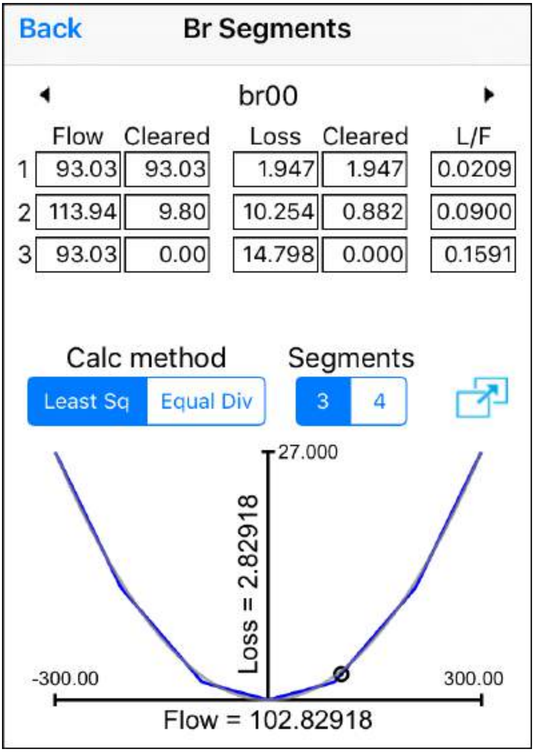

Figure 45: Branch flow-loss segments

In the plot the flow-loss result from the latest solution is indicated by a dot with a circle around it.

The display shows the individual linear segments in blue, with the parabola that they are estimating shown in grey. Using the pinch gesture to zoom in, you can see a close up of the differences between the piecewise curve and the parabola, as shown by the detail in Figure 46.

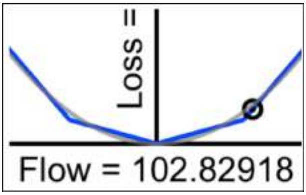

Figure 46: Zoom in to see details of parabola vs segments

To see a full screen version of the plot, as shown in Figure 47, tap the  button.

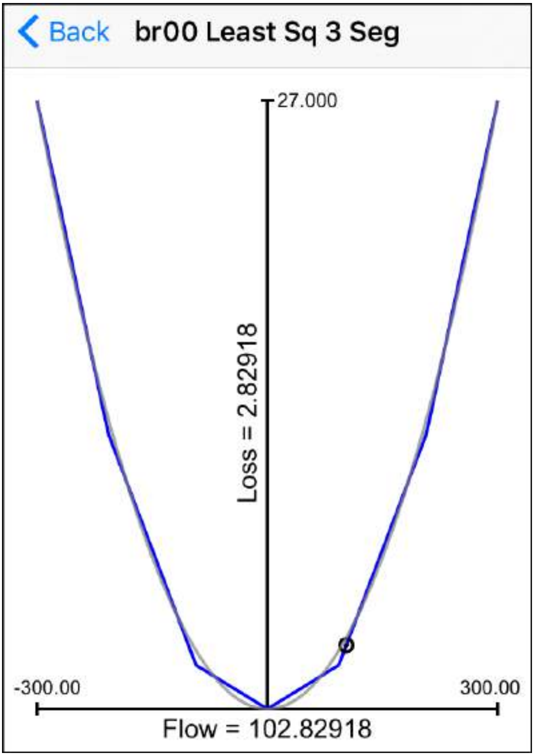

Figure 47: Full screen flow-loss curve

How the segments are calculated

There are different ways of modelling the branch segments. The app allows you to investigate the impact of the following modelling decisions:

- How many branch segments
- What algorithm is used to determine the endpoints of the segments

Number of segments vs total segments

The number of segments refers to the number of segments in each direction, i.e., selecting three segments results in the model using three segments to model flow in the forward direction and three segments in the reverse direction.

Segment endpoints

To determine the endpoints of the segments, the New Zealand electricity market uses an algorithm that minimizes the error between the parabola and the segments. The algorithm is derived from a least squares error function. The app refers to this as the "Least Sq" (least squares) method.

The Singapore electricity market calculates the segments by dividing the maximum flow into equal divisions. The loss point at the end of the segments is taken directly from the parabola. In the app this is referred to as the "Equal Div" (equal divisions) method.

Note: Unless otherwise stated, the tutorial examples all use "Least Sq" and three segments.

Branch constraints to model losses

As shown in Equation 10, the branch flow is constrained to be equal to the sum of the flow on the individual flow-loss segments.

$$Flow_{branch} = \sum_{Segments_{branch}} Flow_{segment}$$

Equation 10: Branch flow is sum of segment flows

The "branch flow is sum of segment flows" constraints can be viewed on the Constraints display, as shown in Figure 48.

```
br00:
BrFlowIsSumOfBrSeg(LTE) constraint:
Shadow Price: $6.92
 +1.00000*br00_{BrFlowPos}
-1.00000*br00_brSeg00_{SegFlowPos}
-1.00000*br00_brSeg01_{SegFlowPos}
-1.00000*br00_brSeg02_{SegFlowPos}  <= 0.00000

br00:
BrFlowIsSumOfBrSeg(GTE) constraint:
Shadow Price: $0.00
 -1.00000*br00_{BrFlowPos}
+1.00000*br00_brSeg00_{SegFlowPos}
+1.00000*br00_brSeg01_{SegFlowPos}
+1.00000*br00_brSeg02_{SegFlowPos}  <= 0.00000

br00:
BrFlowIsSumOfBrSeg(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_{BrFlowNeg}
-1.00000*br00_brSeg00_{SegFlowNeg}
-1.00000*br00_brSeg01_{SegFlowNeg}
-1.00000*br00_brSeg02_{SegFlowNeg}  <= 0.00000

br00:
BrFlowIsSumOfBrSeg(GTE) constraint:
Shadow Price: $0.00
 -1.00000*br00_{BrFlowNeg}
+1.00000*br00_brSeg00_{SegFlowNeg}
+1.00000*br00_brSeg01_{SegFlowNeg}
+1.00000*br00_brSeg02_{SegFlowNeg}  <= 0.00000
```

Figure 48: Constraints display: Branch flow is sum of segment flows

Because the segment constraints link branch flow to segment flows, in order to schedule flow on the branch the solver must schedule flow on one or more of the branch segments…. and in order to schedule flow on a branch segment the solver must also schedule the corresponding loss, due to the constraint shown in Equation 11 which requires that every MW of segment flow has a corresponding loss, determined by the segment's loss-flow-ratio. Figure 49 shows the "loss for flow" constraints for one of the three segments.

$$Loss_{segment} = LossFlowRatio_{segment} \times Flow_{segment}$$

Equation 11: Segment loss-for-flow constraint

Each segment also has its own flow limit, as described by Equation 12, and shown in Figure 50.

$$Flow_{segment} \leq MaxFlow_{segment}$$

Equation 12: Segment maximum flow constraint

This segment flow limit is necessary in order to effectively model the parabola… without it the solver would schedule all of the flow on the segment with the lowest loss-flow ratio.

```
br00_brSeg00:
BrSegLossForFlow(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg00_{SegLossPos}
-0.02093*br00_brSeg00_{SegFlowPos}  <= 0.00000

br00_brSeg00:
BrSegLossForFlow(GTE) constraint:
Shadow Price: $76.92
 -1.00000*br00_brSeg00_{SegLossPos}
+0.02093*br00_brSeg00_{SegFlowPos}  <= 0.00000

br00_brSeg00:
BrSegLossForFlow(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg00_{SegLossNeg}
-0.02093*br00_brSeg00_{SegFlowNeg}  <= 0.00000

br00_brSeg00:
BrSegLossForFlow(GTE) constraint:
Shadow Price: $0.00
 -1.00000*br00_brSeg00_{SegLossNeg}
+0.02093*br00_brSeg00_{SegFlowNeg}  <= 0.00000
```

Figure 49: Constraints display: Segment flow linked to segment loss (one segment shown)

```
br00_brSeg00:
BrSegFlowMax(LTE) constraint:
Shadow Price: $5.31
 +1.00000*br00_brSeg00_{SegFlowPos}  <= 93.03062

br00_brSeg00:
BrSegFlowMax(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg00_{SegFlowNeg}  <= 93.03062

br00_brSeg01:
BrSegFlowMax(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg01_{SegFlowPos}  <=
113.93877

br00_brSeg01:
BrSegFlowMax(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg01_{SegFlowNeg}  <=
113.93877

br00_brSeg02:
BrSegFlowMax(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg02_{SegFlowPos}  <= 93.03072

br00_brSeg02:
BrSegFlowMax(LTE) constraint:
Shadow Price: $0.00
 +1.00000*br00_brSeg02_{SegFlowNeg}  <= 93.03072
```

Figure 50: Constraints display: Each branch segment has its own limit

Options for assigning branch losses

The constraints shown above describe how losses are calculated. There also needs to be some way of subtracting these losses from the branch flow. This is achieved by including the branch losses as an out-flow in the node balance constraint.

The app offers two options for applying the branch losses to the node balance constraint:

- Assign half the branch losses to the sending bus and half to the receiving bus.
- Assign all branch losses to the receiving bus.

The app provides the option of selecting either method, via the solver setting shown in Figure 51.

Figure 51: Solver setting for loss allocation

Bus constraints to model branch losses

If the "Rcv Bus" option for Loss Location is selected then a branch's losses are assigned to the bus that receives power from the branch, and the node balance constraint is as shown in Equation 13.

$$\sum Bus_{FlowIn} - \sum Bus_{FlowOut} - \sum_{\substack{segment \\ FlowIn}} Loss_{segment} = 0$$

Equation 13: Node balance: Losses assigned to receiving bus

If the "50/50" option is selected then the node balance constraint is as shown in Equation 14.

$$\sum Bus_{FlowIn} - \sum Bus_{FlowOut} - 0.5 \times \sum_{\substack{segment \\ FlowIn \\ FlowOut}} Loss_{segment} = 0$$

Equation 14: Node balance: Losses assigned 50/50 to sending bus and receiving bus

Viewed via the app these constraints appear as shown in Figure 52 and Figure 53.

Note: Unless otherwise stated the examples use the "Rcv Bus" setting, i.e., losses are assigned to the receiving end bus. The New Zealand electricity market originally assigned dynamic losses 50-50 but now assigns them at the receiving end.

CONSTRAINTS FOR BUS00

```
bus00:
NodeBalance(LTE) constraint:
Shadow Price: $0.00
 -1.00000*br00_{BrFlowPos}
+1.00000*br00_{BrFlowNeg}
-1.00000*br00_brSeg00_{SegLossNeg}
-1.00000*br00_brSeg01_{SegLossNeg}
-1.00000*br00_brSeg02_{SegLossNeg}
+1.00000*bus00_gen00_offer00_{Cleared}  <=
0.00000
```

```
bus00:
NodeBalance(GTE) constraint:
Shadow Price: $70.00
 +1.00000*br00_{BrFlowPos}
-1.00000*br00_{BrFlowNeg}
+1.00000*br00_brSeg00_{SegLossNeg}
+1.00000*br00_brSeg01_{SegLossNeg}
+1.00000*br00_brSeg02_{SegLossNeg}
-1.00000*bus00_gen00_offer00_{Cleared}  <=
0.00000
```

Figure 52: Bus constraints for losses at receiving end

CONSTRAINTS FOR BUS00

```
bus00:
NodeBalance(LTE) constraint:
Shadow Price: $0.00
 -1.00000*br00_{BrFlowPos}
+1.00000*br00_{BrFlowNeg}
-0.50000*br00_brSeg00_{SegLossPos}
-0.50000*br00_brSeg01_{SegLossPos}
-0.50000*br00_brSeg02_{SegLossPos}
-0.50000*br00_brSeg00_{SegLossNeg}
-0.50000*br00_brSeg01_{SegLossNeg}
-0.50000*br00_brSeg02_{SegLossNeg}
+1.00000*bus00_gen00_offer00_{Cleared}  <=
0.00000

bus00:
NodeBalance(GTE) constraint:
Shadow Price: $70.00
 +1.00000*br00_{BrFlowPos}
-1.00000*br00_{BrFlowNeg}
+0.50000*br00_brSeg00_{SegLossPos}
+0.50000*br00_brSeg01_{SegLossPos}
+0.50000*br00_brSeg02_{SegLossPos}
+0.50000*br00_brSeg00_{SegLossNeg}
+0.50000*br00_brSeg01_{SegLossNeg}
+0.50000*br00_brSeg02_{SegLossNeg}
-1.00000*bus00_gen00_offer00_{Cleared}  <=
0.00000
```

Figure 53: Bus constraints for losses assigned 50/50

Per-unit values and loss calculation

The calculation of power flow is potentially complicated by the fact that different parts of the power system run at different voltages. In order to be able to ignore these voltage differences the simplified power flow equation uses susceptance values that have been adjusted so that they take into account the branch's nominal voltage. This is referred to as using per-unit values. As you may have noticed, the resistance, reactance and susceptance values used by the model are all quoted in per-unit, abbreviated to p.u., e.g., see Figure 54.

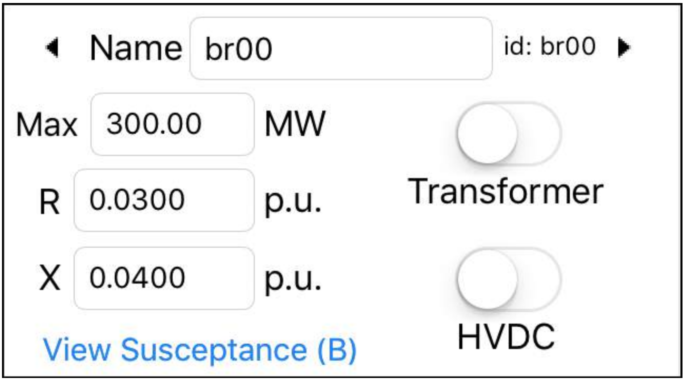

Figure 54: The model uses per-unit values

These per-unit values were calculated prior to being entered into the app. Per-unit values are calculated by dividing the actual values by a base value, e.g., per-unit resistance is calculated as follows:

$R_{per\text{-}unit}=R_{actual}/R_{base}$

The base value depends on the voltage. The base resistance is calculated from a base voltage value and a base power value…

$R_{base}=V_{base}^2/P_{base}$

…where the base voltage is the nominal voltage of the component, e.g., for a branch designed to run at 220kV, the base voltage is 220kV.

To calculate the base resistance there also needs to be a base power value. The base value is arbitrary but the same value must be used consistently in the calculation of all the per-unit values that are provided to a model.

For a power system the per-unit values are commonly specified in terms of a 100MVA base power value. When combined with the typical voltages of a power system, the 100MVA base power value results in per-unit resistance, reactance and susceptance values that are not too big and not too small.

For the power flow calculation where all flows are relative, the fact that the per-unit values are calculated using a 100MVA base can be ignored. However, to calculate the branch losses the 100MVA base will impact the calculations. Therefore, as part of the branch loss calculation the app divides the per-unit resistance value by 100 in order to account for the presence of the 100MVA base in the original per-unit calculation.

If you want to enter per-unit values that did not use a 100MVA base, and you want to model losses, then you will need to convert the per unit values before entering them into the app. For example, if you had per-unit resistance and reactance values that were calculated using a 10MVA base then you would need to multiply their values by 10 before using them in the app.

Transmission rentals

Congestion charges and transmission rentals

In the transmission tutorial we saw that a binding branch can result in the total payments made by the load exceeding the total payments received by the generator. The difference between these amounts is displayed as the $Grid value on the Results display.

Branch losses can also give rise to $Grid, as shown by the results in Figure 55, which are for the single branch model of Figure 43, where the branch is not binding. These $Grid are due to line losses and are referred to as transmission rentals.

Objective	8801.958	Δ 0.000
Iterations	9	Δ 0
Time	0.030 s	Δ -0.105 s
Constraints	32	Δ 0
Variables	50	Δ 0
Gen	102.829	Δ 0.000
Load	100.000	Δ 0.000
Losses	2.829	Δ 0.000
Reserve	0.000	Δ 0.000
$Load	7692.308	Δ 0.000
$Gen	7198.042	Δ 0.000
$Grid	494.265	Δ 0.000
$Reserve	0.000	Δ 0.000

Figure 55: Branch losses give rise to transmission rentals

Non-parabolic loss model has no transmission rentals

The cause of transmission rentals is the parabolic nature of the relationship between branch flow and branch losses. If the branch losses are not parabolic then there are no transmission rentals. We can demonstrate this by editing the single branch model so that the branch only uses the first flow-loss segment; if the solver *only* uses the first segment then the relationship between flow and loss is a straight line.

For the line loss example that we already have (Figure 43), the branch segments in Figure 45 show that the first segment has a maximum flow of 93.03MW. To only use this first segment, edit the load to reduce its quantity from 100MW to 90MW. As shown by the result in Figure 56 the transmission rentals are now $0.

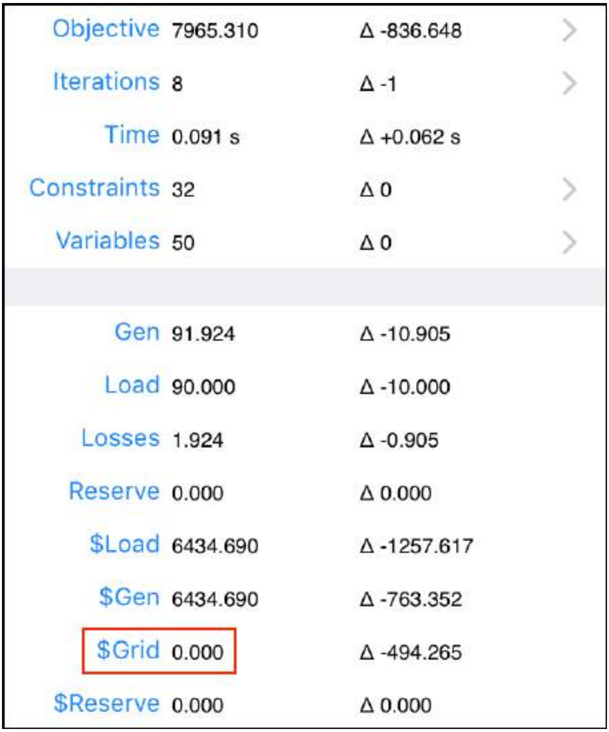

Figure 56: Rentals $0 when only first branch segment used

Why straight line losses have no loss rentals

When only the first branch segment is used, the relationship between flow and losses is a straight line. Although there is a price difference between the load bus and the generation bus this price difference only represents the value of the power that was dissipated as losses; the load pays a higher $/MW price because it is effectively paying for the quantity of power that was generated, but only receiving the power that was delivered. Overall the load pays the same amount as the generator receives and there is no extra payment.

Because the bus price represents the $/MW cost of the *next* MW, with a straight-line loss function the $/MW cost of the next MW is the same as the cost of the scheduled MW, because the losses per MW for the next MW are the same as the losses per MW for the scheduled MW.

Why parabolic losses have loss rentals

The actual physical losses are parabolic. Hence, if we could model a parabola then the $/MW losses associated with the *next* incremental MW would not be the same as the per MW losses associated with the scheduled MW.

As a crude example… with parabolic losses if a scheduled flow of 1MW incurs a loss that is 10% of flow, then we can infer that the resistance is 0.1 per unit, i.e., loss = 0.1 x flow^2. An "incremental" 1MW would take the flow to 2MW and incur a loss of 0.1 x 2^2 = 40%. The bus price is set by the cost of the incremental MW therefore the bus *price* would be set based on a 40% loss, even though the scheduled load is only incurring a 10% loss. The load pays for losses that are not actually incurred, but which would be incurred by the "next" MW. Hence the load pays more than the generator receives.

How the piece-wise parabola causes loss rentals

The LP model cannot include a parabola. But when the piece-wise linear loss model schedules flow that uses more than the first flow-loss segment, it will begin to approximate a parabola, and this will be reflected in the loss rentals.

To see how this happens, we will adjust the model so that it is using more than the first flow loss segment. We could achieve this by adjusting the load again, but it will be more interesting if we increase the number of flow loss segments… on the Branch Segments display for br00, tap the "4" button to increase the number of branch segments from 3 to 4. As shown in Figure 57, the display adjusts immediately to show the new segments and we can see that going to a four-segment model will move the scheduled flow into the second segment.

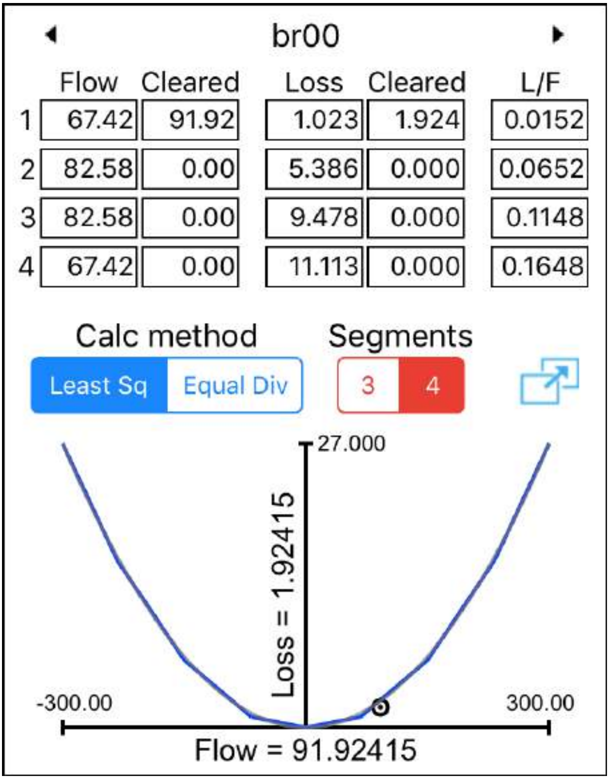

Figure 57: Change to 4 segments (not yet re-solved)

The "Segments" button is highlighted red to indicate that this change has not yet been applied to the model... this will happen when you leave the display via the Back button. Tap the Back button and then tap "OK" when you are prompted to apply the change.

After solving with 4 segments, the flow-loss result is shown in Figure 58 and the corresponding loss rentals are shown in Figure 59.

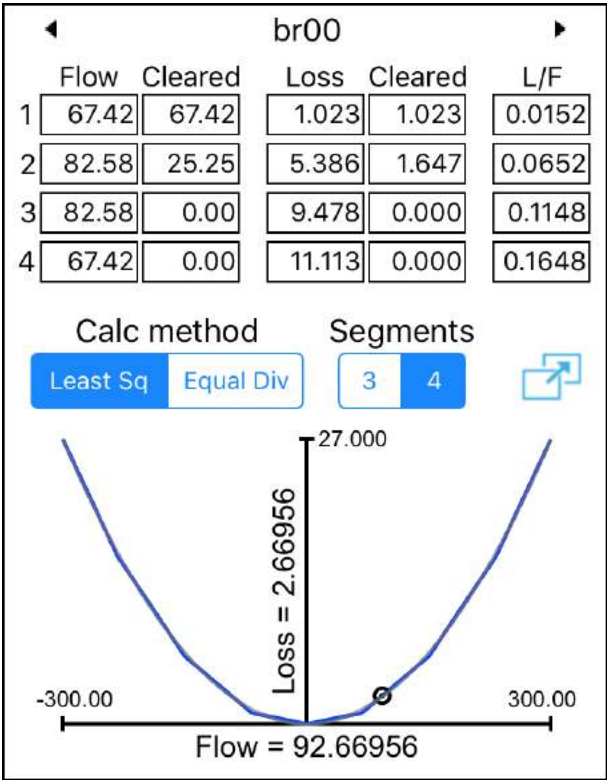

Figure 58: Flow-loss result after solving with 4 segments

Objective	7913.131	Δ -52.179
Iterations	10	Δ +2
Time	0.116 s	Δ +0.025 s
Constraints	38	Δ +6
Variables	60	Δ +10
Gen	92.670	Δ +0.745
Load	90.000	Δ 0.000
Losses	2.670	Δ +0.745
Reserve	0.000	Δ 0.000
$Load	6739.604	Δ +304.914
$Gen	6486.869	Δ +52.179
$Grid	252.735	Δ +252.735
$Reserve	0.000	Δ 0.000

Figure 59: Increasing to 4 segments has brought back the loss rentals

This result includes loss rentals because the load receives the final 25.25MW of its 90MW via the second flow-loss segment. The value of the next MW, and hence the bus price, reflects the flow-loss ratio of the second segment. Hence, the load pays as if all of the 90MW were delivered with this flow-loss ratio.

If all of the 90MW had incurred losses at the rate associated with the second segment, then the payments made by the load would match the payments received by the generation.

However, the first 67.42MW is only subject to the 0.0152 flow-loss ratio of the first segment, compared to the 0.0652 of the second segment that set the price. The generation *quantity*, and hence the amount paid to the generator, reflects the fact that some of the losses were incurred at a lower ratio, but the load pays at a price that was set as if all of the generation quantity was subject to the higher flow-loss ratio of the second segment. Hence the load pays more than the generation receives… with the difference referred to as transmission rentals.

Non-physical losses

Non-physical losses due to negative prices

When branch losses are included in the model, negative prices can lead to non-physical losses. As explained in the Spring Washer tutorial, negative prices occur at a bus where a decrease in available power would benefit the objective value.

One way to decrease the available power at a bus is via load. Another way is via branch losses. Hence, a negative price at a bus indicates that there is a benefit to increasing the branch losses assigned to that bus.

Increasing branch losses

Within the constraints of the model it is possible to increase branch losses because while branch flow is the sum of the branch segment flows, the only thing ensuring that the "correct" segments are used is that usually the solver is trying to minimize the branch losses. Using the "correct" segments means that as the scheduled flow increases, its flow-loss ratio approximates a parabola.

When there are negative prices the solver can use branch segments that maximize the losses. The scheduled flow-loss no longer follows the parabola and the resulting losses are an over-estimate of the actual physical losses that would occur for the scheduled flow.

The losses that are over-estimated are referred to as non-physical losses. The associated result schedules more generation than would actually be required.

Demonstrating non-physical losses

To demonstrate non-physical losses, build the spring washer model shown in Figure 60. Leave all components with their default values except change the limit on br01 to be 20MW. Ensure that branch losses are set to “Least Squares” and 3 segments (via the Branch-Losses display), with losses assigned at the receiving end of the branch (via Solve Settings).

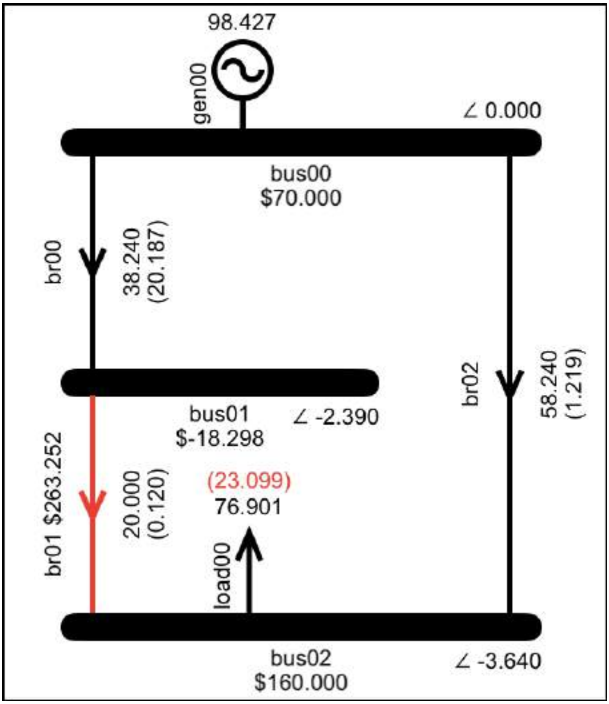

Figure 60: Spring washer with losses

Figure 60 also shows the results. This is the same model that we built in the spring washer example, and again we have negative prices at bus01. What is different is that this time the model includes line losses and the line losses on br00 are 20.187MW.

This level of losses is not physically realistic on a branch that is only transmitting 38.24MW. The loss curve in Figure 61 shows how the solver has achieved the non-physical losses on br00.

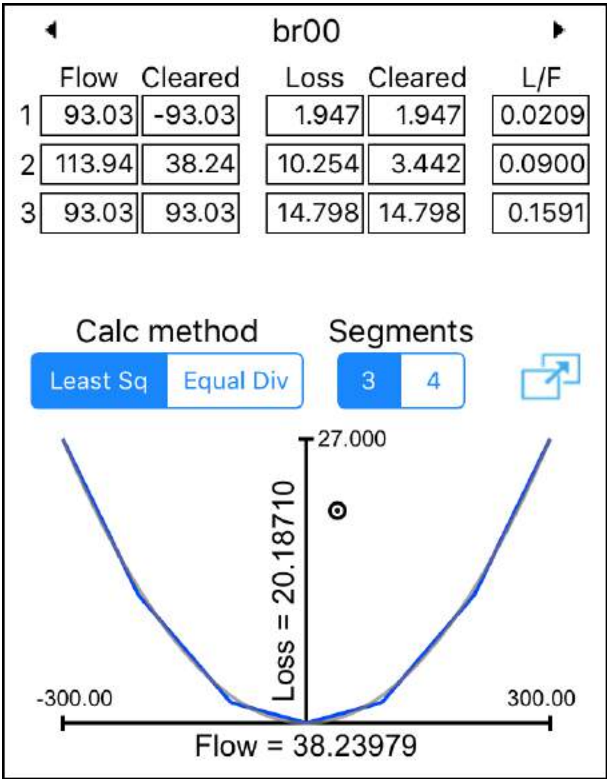

Figure 61: Spring washer with non-physical losses on br00

Because the solver can use any of the segments including those in the reverse directions, it has fully scheduled segment 1 in the reverse direction, balancing this flow by fully scheduling segment 3 in the forward direction… the solver is scheduling circulating branch flows in order to increase the losses assigned to bus01. While the scheduled flow of 38.24 is transferred via segment 2, the losses are the sum of the cleared losses on all three segments.

Branch segments as viewed by the solver

Although the app displays the flow loss curve as a parabola, a graphical view of the constraints is as shown in Figure 62.

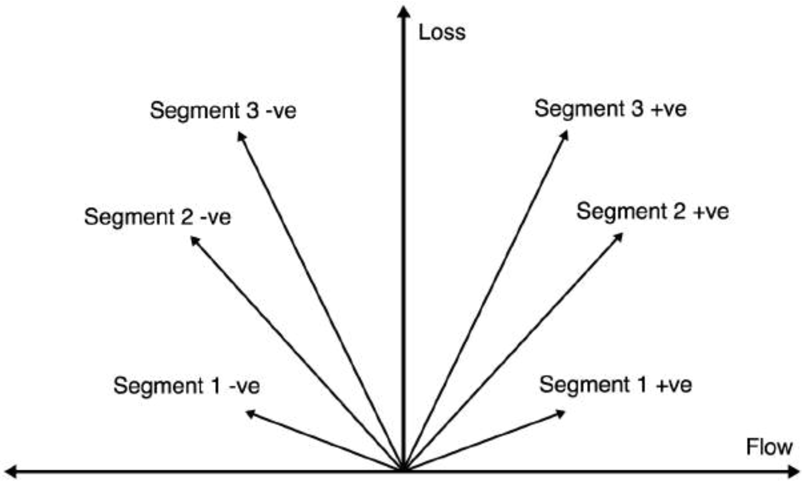

Figure 62: Branch segment constraints

In the normal situation, with no negative prices, the solver is minimising losses and will use the segments with the lowest loss-flow ratio first. The solver will schedule flow on segment 1 to its maximum, then segment 2 and then segment 3… in this way the scheduled flow and loss will track the parabola.

When negative prices incentivise the solver to maximise losses it can utilize circular flows to schedule the segment flows “out of order”, resulting in the total scheduled flow leaving the parabola as we saw in Figure 61.

Removal of non-physical losses

In an actual electricity market, any non-physical losses are detected in post-processing and removed by re-solving the schedule with some of the branch segments removed.

The removal of branch segments from the model is achieved either via post-processing logic which removes the branch segments and then re-runs the solve, or via post-processing logic that re-runs the solve as a mixed integer solution whereby the solver is forced to make binary decisions regarding which segments to include in the solution.

Summary

In this section we demonstrated how parabolic transmission losses are modelled in a linear solver by using a piece-wise linear curve.

We explained why the modelling of transmission losses gives rise to transmission rentals, whereby the amount paid by the load is greater than the amount paid to the generator.

We demonstrated that negative prices have the potential to cause the solver to produce circulating branch flows that result in non-physical losses, and we explained how and why this happens.