9th International Conference on Hydroinformatics
HIC 2010, Tianjin, CHINA

ENHANCING THE ROBUSTNESS OF WATER RESOURCE SIMULATION MODELS BASED ON NETWORK LINEAR PROGRAMMING

GEORGE KUCZERA[1,2], LIJIE CUI[1,2], RACHEL GILMORE[2,3], ANDREW GRADDON[1,2], S. MOHAMMAD MORTAZAVI N[1,2]

[1]*School of Engineering, University of Newcastle, Australia*

[2]*eWater CRC, Canberra, Australia*

[3]*CSIRO, Land and Water, Canberra, Australia*

In Australia, water resource simulation models using network linear programming (NLP) are widely used. This study identifies two potential problems when using NLP models at daily time steps, which, if not addressed, could result in poor outcomes. These problems are the failure of iteration to converge to good solutions and suboptimal water allocations due to a lack of foresight. It is shown how enhancements to the WATHNET model have addressed these issues. The outcome is a more robust framework than provided by the current generation of Australian NLP-based models.

INTRODUCTION

In Australia, water resource simulation models are widely used to evaluate the performance of regulated river basins to support urban, irrigation and environmental needs. Two modeling approaches are typically used to undertake such simulation: rule-based models [e.g., IQQM (Simons *et al*., 1996), MSM-BIGMOD) and objective-driven models (e.g., REALM (Perera and James, 2003), WATHNET (Kuczera, 1992)]. The objective-driven models are fundamentally different from rule-based models. At their core is a network linear program (NLP) which uses a hierarchy of objectives to assign water within a network. Both modeling approaches have their advantages. Rule-based models typically have faster run times and produce results that can be readily explained to stakeholders, whereas objective-driven models are far less complex to setup and can produce more efficient water allocations at the expense of longer runtimes (Gilmore *et al*., 2009) and less transparency about how allocations are made.

This paper focuses on NLP-based simulation models. Current practice in Australia has seen NLP-based applications mainly applied using monthly time steps. However, for a variety of reasons, there is a growing need to apply such models at daily steps. This paper considers a range of potential problems which, if not addressed, could result in poor outcomes. In particular, two shortcomings of NLP-based models are identified, namely failure of iteration to converge to good solutions, and suboptimal water allocations due to a lack of foresight. It is shown how enhancements to the WATHNET model have addressed these shortcomings.

CONVERGENCE FAILURE OF NLP ITERATION

A pure NLP solves the following problem:

$$\begin{aligned} &\min c^T x \\ &\text{subject to } Ax = b,\ l \le x \le u \end{aligned} \tag{1}$$

where x is a vector of arc flows, c is a known vector of costs, A is a node-incidence matrix describing the connectivity of the network, b is a vector of known nodal demands or supplies, l is a vector of known minimum arc flows (usually zero), and u is a vector of known maximum arc flows. The equality constraints ensure a mass balance at each node, while the inequality constraints ensure arc flows remain between known minimum and maximum capacities.

In practice, the complexities of water resource management require implementation of constraints which cannot be accommodated within a pure-NLP framework. This problem is traditionally dealt with using iteration in which successive NLPs are run with progressively adjusted values for the maximum capacity u until convergence occurs. While this strategy works well for NLP models using monthly time steps, a shift to daily time steps is likely to introduce complexities which iteration cannot adequately handle. Ilich (2008) documents a number of common scenarios where iteration with NLP models will fail to produce a good solution. What is particularly worrisome is that the NLP iteration can converge to a solution, which unfortunately is far from efficient.

One of the main modeling issues encountered when going from monthly to daily time steps is the need to more explicitly account for hydraulic constraints. The following example illustrates a typical failure of NLP iteration in the presence of hydraulic constraints. Figure 1a shows a river network with two reservoirs, Split Rock and Keepit, in series – this is an idealization of a real network in New South Wales, Australia. A demand node is located downstream of the lower reservoir, Keepit. In this hypothetical example, the head-discharge characteristics of each reservoir result in a hydraulic constraint where the maximum daily volume of water that can be released from each reservoir is 10% of the stored volume. Because a large unregulated tributary flows into Keepit, the most efficient storage strategy is to preferentially store water in Split Rock and draw down Keepit. This maximizes the air space in Keepit for capturing unregulated tributary inflow.

Figure 2 presents the daily time series trace for Split Rock and Keepit storage volumes, the demand at node 7 and the shortfall in supplied demand produced by an iterative NLP simulation. In the iterative scheme, the maximum release from each reservoir is based on the storage volume in the previous iteration or, in the case of the first iteration, on the storage volume at the end of the previous day. Around day 70, shortfalls in demand start. By about day 90, the shortfalls are close to 100% as the storage volume in Keepit virtually empties. What is striking is that there is a large stored volume in Split Rock reservoir. The NLP never released water from Split Rock

during the shortfall period, even though there was an abundance of water.

To understand this sub-optimal behaviour recall that the NLP was assigned higher carryover gains to Split Rock than to Keepit to ensure Keepit had air space to harvest unregulated tributary inflows. Therefore the NLP would only release water from Split Rock to avoid shortfalls at downstream nodes.

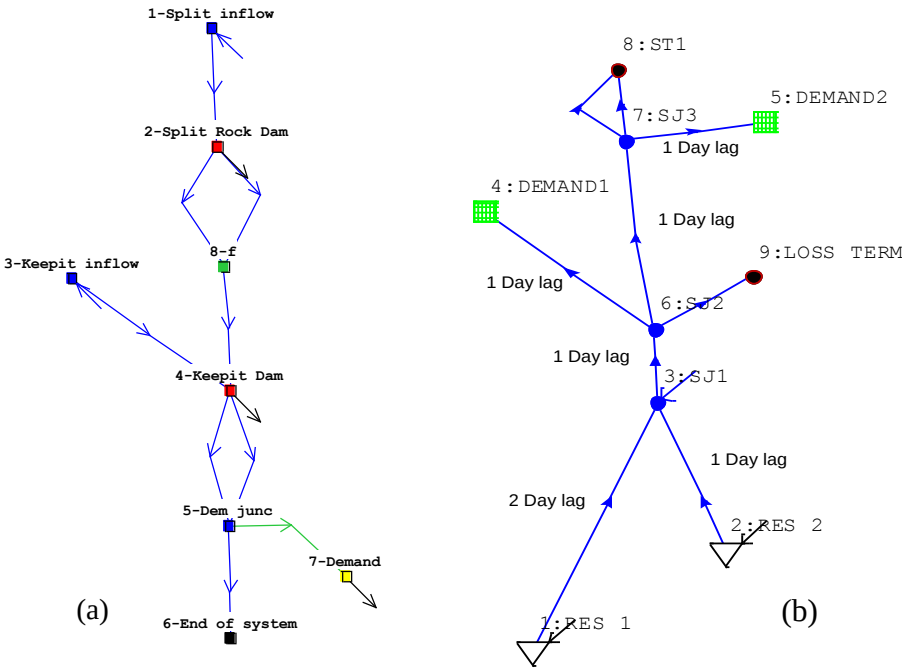

Figure 1. River basin schematics: (a) Split Rock-Keepit system; (b) Sample river basin with travel times

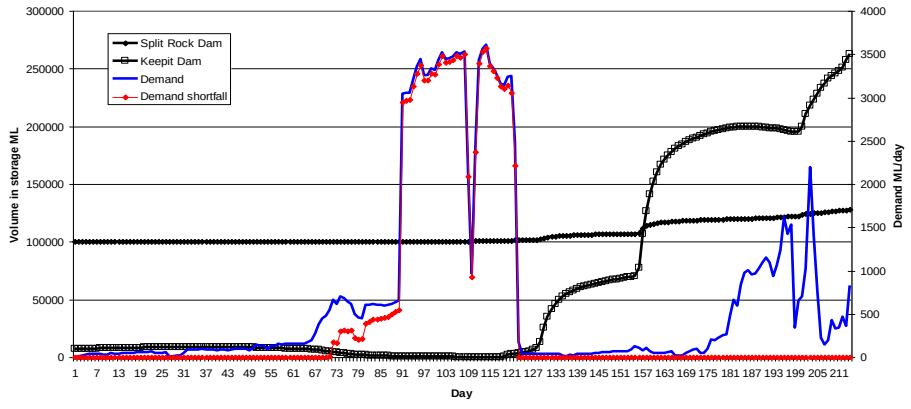

Figure 2. Daily time series traces for Split Rock and Keepit storage volumes, node 7 demand and shortfall in supplied demand produced by an iterative NLP simulation

In the iterative solution the maximum daily release from each reservoir was set at the fixed value equal to 10% of the storage volume in the previous iteration. Because Keepit storage was low the maximum daily release was insufficient to meet demand from about day 70. The optimal strategy would have been to release water from Split Rock to increase the storage at Keepit so that its outlet could pass sufficient water to meet demand. Unfortunately the NLP could not "see" this strategy because the maximum release from Keepit was fixed. Releasing water from Split Rock was a "sub-optimal" strategy because the water could not meet the demand shortfall (on account of the fixed release capacity from Keepit) and water stored in Keepit received a smaller gain than water stored in Split Rock. No amount of iteration can change this fundamental shortcoming.

The only robust way of overcoming this shortcoming is to make the linear program "see" that the daily release capacity is dependent on storage. This can be simply implemented using a general linear program (LP) rather than a NLP. A LP supports any linear constraint as part of its formulation. However, LP solvers are about two orders of magnitude slower than NLP solvers. Fortunately, there exists a specialized LP algorithm called network linear program with side constraints (NLP-SC) (see Castro and Nabona, 1994) which is formulated as follows:

$$\begin{aligned} &\min c^T x \\ &\text{subject to } Ax = b,\ l \le x \le u,\ e \le Dx \le f \end{aligned} \tag{2}$$

where $e \le Dx \le f$ defines an additional linear constraint with e and f known vectors and D a known matrix. This algorithm is well suited to water resource simulation where the majority of constraints take the form of nodal mass balance and fixed arc flow capacities and only a limited number of constraints involve a linear combination of the form $e \le Dx \le f$. It exploits this fact to produce solution times significantly closer to those of a pure NLP than a general LP.

Figure 3 presents the solution found using NLP-SC where the hydraulic constraints on reservoir releases were expressed as side constraints. What is immediately evident is that NLP-SC recognized the need to transfer water from Split Rock to Keepit to provide sufficient head to enable releases that reduced demand shortfalls. One can see that Split Rock was drawn down to virtually an empty state while Keepit was maintained at a sufficient level so that significant releases of water could be made to the demand node. The use of side constraints restored the ability of the NLP to "see" the whole system when making allocations.

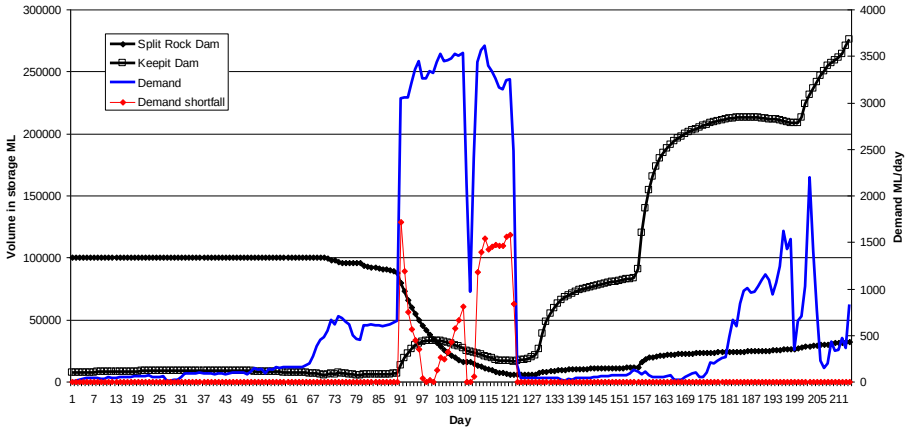

Figure 3. Daily time series traces for Split Rock and Keepit storage volumes, node 7 demand and shortfall in supplied demand produced by a NLP with side constraints simulation

SUB-OPTIMAL ALLOCATION DUE TO A LACK OF FORESIGHT

Figure 3 highlights another limitation of standard NLP formulations. At about day 90, the NLP "suddenly" finds itself unable to meet daily demand. This has occurred because the NLP has no foresight of future demand and inflows. At monthly time steps, this issue is not as apparent and confronting as at daily time steps. In the case of Figure 3, assuming there is knowledge of future demand with lead times up to a week, the preferred operating policy would commence releases from Split Rock so that Keepit accumulates sufficient storage to provide sufficient head to ensure releases meet future downstream demand.

Fortunately there is a simple way to introduce foresight. Kuczera (1989) showed how this could be done for networks for which travel time is not a consideration. However, at the daily time step, in most Australian irrigation systems, a release from an upstream reservoir will not necessarily arrive at a demand node in the same day. It is therefore necessary to generalize Kuczera's (1989) formulation. This generalization is best understood by considering an example. Figure 1a presents a simple network with two reservoirs and two demands linked by a stream network with travel times expressed in days – for example, a release from the reservoir at node 1 on day t would arrive at the junction node 3 on day t+2. The inclusion of foresight and travel time (assumed independent of flow) can be done exactly using the NLP graph shown in Figure 4.

Foresight is implemented by repeating the network multiple time steps into the future. At each time step the network receives known nodal inputs corresponding to that time step. Storage is carried over between time steps using carryover arcs, denoted by the blue dashed arcs in Figure 4. For example, using the notation (i,j) to denote an

arc flowing from node *i* to node *j*, storage at node 1 at the end of the current time step is conveyed by arc (1,13) to node 13 to define the storage at the start of the next time step.

Making provision for integer travel times can be done exactly in a NLP by using carryover arcs in a fashion similar to that used to implement foresight. Again this concept is best explained by an example. The travel time along (1,3) is 2 days. Referring to Figure 4, the black dashed arcs represent arcs with non-zero integer travel times. These arcs are removed and replaced by carryover arcs. Thus arc (1,3) is removed and replaced by two carryover arcs (1,24) and (24,37) – node (24) is not necessary but simplifies the automatic generation of the NLP. A release from the reservoir at node 1 on day t will therefore not arrive at node 3 on day t but at node 37 on day *t*+2. The extension to travel times expressed as a fraction of day is straightforward. For example, if the basic travel time unit was 0.25 day, a travel time of 2 days would require carryover over eight 0.25-day time steps.

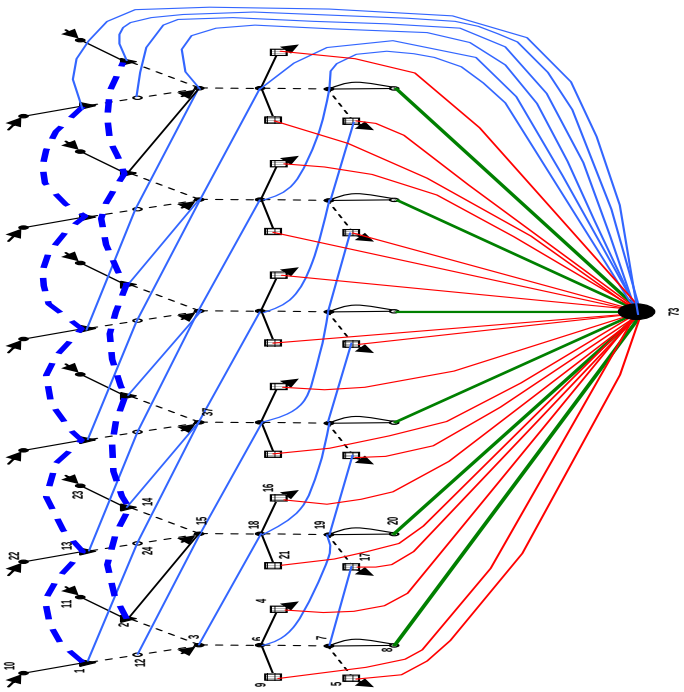

Figure 4. NLP graph for Figure 2a river basin depicting foresight and provision for travel times

Figure 5 presents the solution found using NLP-SC with 7-day foresight. Comparing it with Figure 3 one can see that the foresight NLP, "seeing" the future sharp rise in demand at node 7, commences to release water earlier from Split Rock so that at day 90 there is sufficient water in Keepit to meet downstream obligations. As a result, the short but sharp shortfall shown in Figure 3 around day 90 is avoided. Of

course by about day 100 Split Rock has been depleted to such an extent that it is no longer possible to meet full downstream commitments. The important point is that this shortfall is not due to a shortcoming in the NLP but an unavoidable consequence of low storage levels and constrained outlet hydraulics.

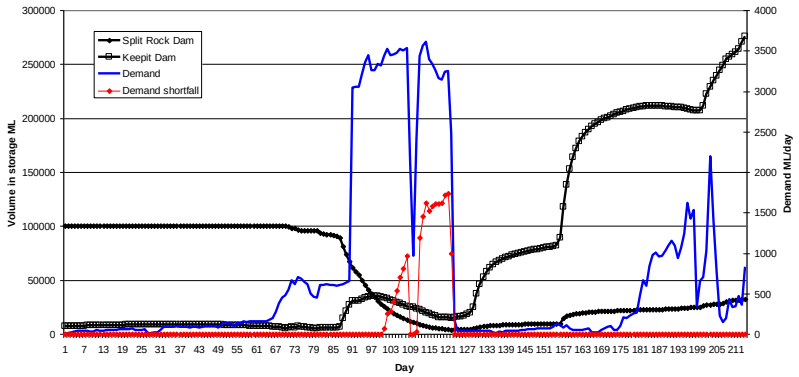

Figure 5. Daily time series traces for Split Rock and Keepit storage volumes, node 7 demand and shortfall in supplied demand produced by a NLP with side constraints simulation and 7-day foresight

It should be apparent that the foresight NLP has application in irrigation systems where reservoir releases must be planned to meet future water orders in the lower part of the system. In addition to lag routing, it can be shown that with the aid of side constraints exact linear storage routing can be performed.

CONCLUDING REMARK

This paper has focused on two shortcomings of the current generation of NLP-based simulation models used in Australia. Unless addressed, these shortcomings may become problematic in NLP applications at the daily time step. The use of iteration in NLP may fail when more explicit hydraulic constraints are introduced. An example is presented showing how iteration fails to produce sensible results when storage-dependent reservoir releases are invoked. It is shown that using a NLP with linear side constraints algorithm can overcome such a problem without the computational penalty that would arise if a general LP solver were used. Another shortcoming arises from the lack of foresight about future demands. It is shown how NLPs can be formulated to "see" ahead and also simulate travel times in large systems.

Two other significant shortcomings associated with NLP-based simulation models are discussed in Kuczera *et al*. (2009). One of the misconceptions about the NLP-based simulation models is that they optimize the performance of the system. In truth, there are many operating and infrastructure variables exogenous to the NLP that affect system performance. For example, gains on reservoir carryover arcs control the

distribution of water across multiple storages – these gains must be specified prior to running the NLP. Likewise, restriction rules affect the actual demand submitted to the NLP – these are again implemented prior to running the NLP. Furthermore, infrastructure variables such as reservoir capacity and treatment plant capacity are typically defined prior to running the NLP. Mortazavi *et al*. (2009) show how embedding the NLP-based simulation model in a multi-objective optimization framework overcomes this limitation. Another limitation arises in the evaluation of drought security risks in highly reliable urban systems, a task that becomes computationally prohibitive when using daily time steps. Cui and Kuczera (2009) show how the use of a critical period compression strategy can very substantially reduce computational effort in such cases.

REFERENCES

[1] Castro, J. and N. Nabona "An implementation of linear and nonlinear multicommodity network flows", *European Journal of Operational Research*, Vol. 92, (1996), pp 37-53.

[2] Cui, L. and Kuczera G., "Assessment of the Replicate Compression Heuristic to Improve Efficiency of Urban Water Supply Headworks Optimization" , *J. Water Resources Planning Management*, ASCE, 135(6), (2009), pp 451-457.

[3] Gilmore, R.L., Kuczera, G., Penton, D., and Podger, G., "Improving the efficiency of delivering water in Australian river systems: modelling multiple paths", *18th World IMACS / MODSIM Congress*, Cairns, Australia 13-17 July 2009.

[4] Ilich N., "Shortcomings of linear programming in optimizing river basin allocation", *Water Resour. Res*., pp 44, W02426, doi:10.1029/2007WR006192.

[5] Kuczera, G., "Fast multi-reservoir multi-period linear programming models", *Water Resources Research*, 25(2), (1989) pp 169-176.

[6] Kuczera, G., "Water supply headworks simulation using network linear programming". *Advances in Engineering Software*. 14, (1992) pp 55-60.

[7] Kuczera, G., Cui, L., Gilmore, R., Graddon, A., Mortazavi, M. and Jefferson, C., "Addressing the shortcomings of water resource simulation models based on network linear programming", *Proc. 31st Hydrology and Water Resources Symp*., Newcastle, (2009).

[8] Mortazavi, M., Cui, L. and Kuczera, G., "Application of multiobjective optimization methods for Urban Water Management: a case study for Canberra water supply system", *Proc. 31st Hydrology and Water Resources Symp.*, Newcastle, (2009).

[9] Perera, B.J.C. and James. B. A., "Generalised Water Supply Simulation Computer Software package", *Hydrology Journal, Institution of Engineers* (India), 26/1-2, (2003), pp 67-83.

[10] Simons, M., G. Podger, and R. Cooke, "IQQM; A hydrologic modelling tool for water resource and salinity management", *Environmental Software*, 11, pp 185-192.