Proposals for Improvements to Science and/or Software in APSIM

Instructions to Developers and Reviewers

Proposals for changes to science and/or software in APSIM should be evaluated using four guiding principles:

- Scientific rigour
- Prediction accuracy on test data sets
- Prediction generality beyond test data sets
- Speed, maintainability, and ease of use of code

Scientific rigour

Enhancements to processes or modules should be based on awareness of existing approaches and underpinning science via reference to existing literature and models. The intent of the proposed change should be clearly stated and the basis in science outlined.

Where the requirements of scientific rigour permit, changes to the representation of processes should proceed by generalization (i.e. the previous representation of the process should be recoverable by appropriate parameterization of the model). Adherence to this principle reduces the impact of changes to APSIM on users and their applications.

Prediction accuracy

Simulations on test data sets should be provided with submissions for changes to APSIM. This might involve (i) simulations of crop and/or soil state over time showing simulation trace through time series of data including error bars on observed data, and (ii) comparisons of observed vs. predicted for a range of variables across data sets using appropriate regression testing. Comparisons of simulations with and without the proposed change are required where possible.

Prediction generality

As detailed test data sets are often limited in their range (genetic (G), management (M), and environment (E)) simulations to demonstrate prediction generality of the proposed change (i.e. sensibility analysis/plausibility) should be submitted. This might include simulations across a range of G, M, E factors to demonstrate the capacity of the change proposal to simulate anticipated or known responses.

Coding

Proposed changes should not have a detrimental effect on APSIM run-time speed, be programmed in a manner that clearly shows the intent of the code (e.g. liberal use of comments) and maintains or improves ease of use. If changes are made to the input data requirements of the source code (e.g. parameters), then backwards compatibility must be maintained (e.g. through the use of file converters or optional reading of new parameters)