APSIM Next Generation: The final frontier?

Dean Holzworth [a], N. I. Huth [a], J. Fainges [a], N. I. Herrmann [a], E. Zurcher [a], H. Brown [b], V. Snow [c], S. Verrall [a], R. Cichota [c], A. Doherty [d], P. deVoil [d], G. McLean [e] and J. Brider [e]

[a] CSIRO Agriculture, Australia
[b] Plant and Food Research, New Zealand
[c] AgResearch, New Zealand
[d] University of Queensland, Australia
[e] Queensland Department of Agriculture and Fisheries
Email: dean.holzworth@csiro.au

Abstract: For twenty four years, the Agricultural Production Systems sIMulator (APSIM) has grown from a farming systems framework used by a small number of people, into a large collection of models used by many thousands of modellers internationally. The software consists of many hundreds of thousands of lines of code in 6 different programming languages. The models are connected to each other using a 'common modelling protocol'. This infrastructure has successfully integrated a diverse range of models but isn't capable of easily meeting the challenges outlined above. For these reasons, the APSIM Initiative has begun developing a next generation of APSIM (dubbed APSIM Next Gen) that is written from scratch and designed to 'run anywhere'.

The new framework incorporates the best of the APSIM 7.7 framework with an improved supporting framework. C# was chosen as the programming language and together with MONO, the models and user interface run on Windows, LINUX and OSX. The Plant Modelling Framework (a generic collection of plant building blocks) was ported from the existing APSIM to bring a rapid development pathway for plant models. The user interface paradigm has been kept the same as the existing APSIM version, but completely rewritten to support new application domains and the newer Plant Modelling Framework. The ability to describe experiments has been added which can also be used for rapidly building factorials of simulations. The ability to write C# and VB.NET scripts to control farm and paddock management has been retained. Finally, all simulation outputs are written to an SQLite database to make it easier and quicker to query, filter and graph outputs.

The software engineering process has also been significantly improved. We have adopted GitHub to host the APSIM Next Gen. repository and have built a workflow around it involving feature branches, pull requests for peer-review of code and science reviews for major tasks. We have improved the testing regime and are building validation data sets for all models. These datasets are re-generated every time there is a change to APSIM and regression statistics are compared with previously accepted values. This improves the likelihood of detecting unexpected changes to model performance when a developer commits new changes. We have also enhanced the way we document all models by auto-generating all documentation from the validation tests and from using reflection to examine comments in the source code. The result is a nicely formatted PDF that describes a model and presents its validation, with regression statistics, graphically.

This paper explores each of the design decisions outlined above and discusses why the decision was made to 'start from scratch'.

Keywords: *APSIM, agricultural modelling, model*

1. INTRODUCTION

APSIM (Holzworth *et al.* 2014) is a farming systems model that is used by researchers to simulate a wide range of complex systems. It contains interconnected biophysical and management models to simulate systems comprising soil, crop, tree, pasture and livestock processes. Agricultural production systems modelling has expanded in scope over the last decade (Holzworth *et al.* in press) and as a result, APSIM has evolved beyond the point based, production systems that it was built for in the 1990s. APSIM is now being used for:

- farmer advice (Carberry *et al.* 2002; Hochman, van Rees, *et al.* 2009; McCown *et al.* 2009),
- resource use and efficiency (Hochman, Holzworth, *et al.* 2009),
- plant breeding (Chapman *et al.* 2003; Hammer *et al.* 2010; Messina *et al.* 2011),
- climate change and adaptation (Wang *et al.* 2011; Teixeira *et al.* 2012; Thorburn *et al.* 2012) ,
- livestock and mixed crop-livestock systems (Bell *et al.* 2009; Lilley and Moore 2009; Moore *et al.* 2009),
- food security (Carberry *et al.* 2013),
- yield gap assessments (Hochman *et al.* 2012; van Rees *et al.* 2014),
- whole farm modelling approaches (Snow *et al.* 2014; Rodriguez *et al.* 2014),
- agroforestry systems (Huth *et al.* 2002, 2014),
- horticultural cropping systems (Huth *et al.* 2009; Brown *et al.* 2011),
- carbon sequestration in agricultural soils (Luo *et al.* 2011, 2014),
- biotic and abiotic system constraints (Whish *et al.* 2007).

In addition to the expanding scope, the computing landscape has also changed significantly. The development of APSIM began in 1991 (in FORTRAN) and in the intervening 24 years many scientists and software engineers have extended and redeveloped parts of it, changing computer languages for many of the models and infrastructure. This has resulted in:

- a source code base that is complex and difficult to maintain,
- a framework that isn't fully cross platform,
- slow runtimes,
- documentation that hasn't been kept up-to-date,
- non-existent validation for some models,
- a model development process that is made difficult due to inadequate testing systems and untidy code.

Computing hardware and software has changed from desktop based systems to mobile phones and tablets, web and cloud based portals with web services offering a way to connect the various systems. While iterative development has allowed the APSIM community to evolve the framework for 24 years, a decision was made recently to start development of a replacement system that is built for the expanding scope and resolves many of the deficiencies listed above. The remainder of this paper discusses many of the design decisions and describes the development of 'APSIM Next Generation' (APSIM Next Gen).

2. REQUIREMENTS AND DESIGN DECISIONS

Cross platform development was a key requirement from the outset. The ability to run the APSIM models and user interface on Windows, LINUX and OSX desktop and cluster architectures was important. In addition, with the advent of mobile platforms and web based development, the requirement for running the models (not the user interface) behind web pages and on mobile devices is also important. To achieve this, there are many possible strategies but with the skills of the software team being predominately .NET based, the decision was to adopt C# and use MONO for the cross platform development. The single language decision negates the need for building language adaptors but has the downside of potentially requiring reskilling of some model developers. Java was also considered as an obvious approach but this would have required reskilling the team. No comprehensive cost-benefit analysis was done on different languages and frameworks. Ultimately, the decision was made to continue using the tools and languages that we were already using. While C# and MONO are considered by some to be a niche market, we justify the choice by having experience in using this combination in a range of software tools, on different platforms, in different configurations. To some extent we know the technology and understand its limitations. The decision was also made to support multiple development IDE's (Visual Studio, Sharp Develop and Mono Develop), allowing developers to work on their operating system of choice.

To address the requirement for increased runtime speed, a philosophy of 'keeping it simple' was adopted. Minimising the amount of code that the infrastructure needed to execute would directly improve the runtime speed. Much of the existing runtime of APSIM (version 7.7 and earlier) is spent on the inter-model protocol of

passing variables and events between models. Each individual variable value passed between models (of which there are many dozens each daily time step) is packed and unpacked to and from a binary structure. This strategy is necessary for managing the multi-language nature of the existing APSIM, but isn't necessary for APSIM Next Gen. Instead, models in the new framework directly call methods and properties of other models in the normal object oriented way. To overcome the tight coupling of models, a link mechanism was developed allowing models of the same type (e.g. two water balance models) to be interchanged without requiring changes to the other models that call them (Holzworth *et al.* 2010).

The Plant Modelling Framework (PMF), developed by Brown *et al.* (2014), was adopted and extended from APSIM 7.7. This framework allows plant model developers to more quickly construct plant models from smaller building blocks, often without any coding, allowing scientists to become involved with plant model development. An IDE for clicking together individual plant processes and then parameterising them is available directly from the user interface. Figure 1 shows a plant model developer working on the maize model in the user interface, parameterising the 'Maximum Potential Grain Size' variable of the 'Grain' organ. Processes can be swapped in and out, allowing for very different types of plant models to be constructed.

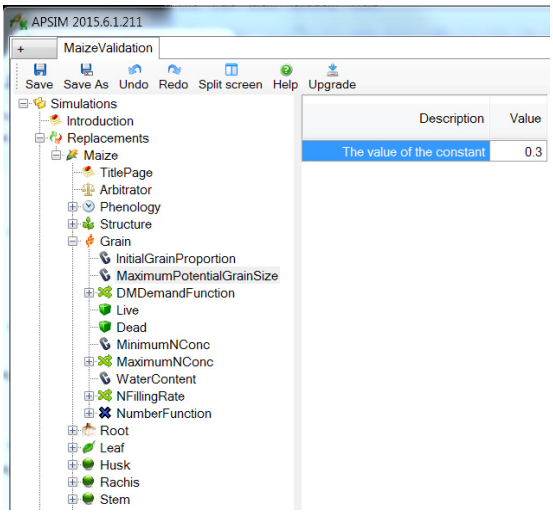

Figure 1. This figure shows the maize model under development in the user interface.

The existing user interface in APSIM 7.7 is generally accepted as being easy to use, intuitive and sufficiently flexible for a wide range of scenario analyses. The same look and feel was adopted in APSIM Next Gen., with a tree control showing the hierarchical nature of simulations (the models) and a right hand panel showing the properties of the selected model. Each tree node in the tree view represents an object in memory. Changing properties on the right changes the properties of the object in memory. This is done via a model-view-presenter pattern. A view is what is visible on the screen, a model is the object in memory (e.g. soil, plant, leaf, photosynthesis – many different levels of granularity) and the presenter sits between the two. The presenter retrieves model variable values and populates the view appropriately. This decoupling of models from their view allows flexibility in the creating of different views (user interfaces) in the future. Web or mobile device based user interfaces are possible without having to change the models.

One of the strengths of APSIM 7.7 is the ability to write scripts that mimic on-farm management practices. This has been retained and extended. Users can write scripts in C# or VB.NET (that get compiled at run-time by the .NET framework) that sow and harvest crops, perform tillage actions, fertilise and irrigate etc. Scripts can also be used to control the user interface (load files, click on models etc). The plan is to allow scripting outside of simulations to enable a simulation to be run multiple times in optimising strategies or climate change scenarios.

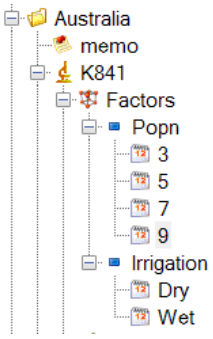

Figure 2. This figure shows a factorial with two treatments population and irrigation rate mimicking an experiment conducted at Katherine, Australia.

An experiment specification system has been constructed allowing users to quickly construct a number of simulations (from a base simulation) without having to specify each individual simulation. Figure 2 shows an experiment configuration of two factors, population (3, 5, 7, 9 plants/m^2) and irrigation rate (wet and dry). The wet and dry irrigation treatments define two different irrigation strategies. This experiment facility can also be used for factorial type simulations where a large number of simulations can be constructed quickly to test the response of a model.

In APSIM 7.7, outputs from a simulation are written to a text file. While this is simple, there are some downsides. For a large number of simulations, there will be a large number of outputs files, which as well as being cumbersome, can be slow to read and parse. In APSIM next generation, all outputs for a group of simulations are written to an SQLite database. This makes it very quick for reading, filtering,

querying and graphing outputs in the user interface. The benefits of text files (easy to read in a text editor and using a diff. tool to compare two files) has been realised by an export option from the database. This strategy offers the best of both worlds.

3. A LITTLE BIT OF SOFTWARE ENGINEERING PROCESS

Given the APSIM development team (approximately is distributed geographically, in different time zones, there was a need to adopt a rigorous software engineering process that was simple but robust. We have been greatly influenced by the Agile Software Manifesto (http://www.agilemanifesto.org/) and the techniques outlined by Jeffries *et al.* (2001) and the SCRUM Manifesto (http://scrummethodology.com/). In particular, we have tried not to over-engineer the software, we adopt very short iteration and release cycles and we use pair programming when needed. Planning and timing of what is implemented is completely demand driven, usually by project milestones that require a particular feature. Most of the team members also use APSIM in various projects (i.e. they are APSIM users as well as software developers) and so the individual team members largely determine what gets implemented. The result is a team that is self-managing to a large extent where all members have control over what gets implemented and released.

We have adopted GitHub (https://github.com/APSIMInitiative/ApsimX) as our version control repository and applied an off the shelf product Jenkins (https://jenkins-ci.org/) as our continuous integration system. The workflow is best described as a feature branch workflow within the GitHub forking model. The workflow is described as follows:

- All releases of APSIM come from the *master* branch on GitHub.
- A developer who wants to make a change to APSIM, 'forks' or clones the APSIM repository into their own GitHub account. They then create a branch (derived from the master branch), give it a name that is appropriate to their work, implement their changes, and commit as often as they need to.
- Once they are happy with their work, they push to their repository on GitHub and raise a pull request. This is a signal that they would like their changes to be peer-reviewed and automatically tested by Jenkins.
- If the change is minor, a software engineer will examine the changes.
- If the change is a major science change then it will be peer-reviewed by a researcher in a similar way to a peer-reviewed journal article.
- If peer-review and the automated testing are satisfied, then the branch is merged with the master branch in the APSIM repository.
- The continuous release system picks this up and makes it available to all users as an upgrade.

All models have validation simulations containing numerous graphs showing the model performance against the observed data. The Jenkins testing system will run the validation simulations, generate validation statistics and compare these against expected statistics. There are also unit tests and sensibility tests where there is a lack of observed data for a particular model (Holzworth *et al.* 2011). This level of testing helps to detect unintended changes to a model's performance and test for stability issues.

All documentation is auto-generated from the validation simulations and by using reflection to examine the source code, in particular the comments on classes and properties. For plant models, the PMF separates the design of a plant model into smaller units allowing each of these to be described from the source code. The result is a clean PDF that describes the model (including graphs of many of the individual functions) and the validation datasets and graphs. Examples can be found here:

https://www.apsim.info/Documentation/APSIM(nextgeneration)/Modeldocumentation.aspx

A continuous release system has been constructed that makes new versions of APSIM immediately available to users. When developers make a change to APSIM and the change is peer-reviewed and accepted, a new release is automatically created and pushed to users. During high development periods, this can happen many times per day. The advantage is very quick turnaround times for defect fixes but sometimes new defects are created (through insufficient testing) and pushed out to users which isn't desirable. Users have the choice of whether they wish to upgrade. This greatly shortens the time between when an issue is identified to when it is delivered to users. It also has the advantage of providing statistics on APSIM usage as all upgrades are recorded in a database.

4. DISCUSSION AND CONCLUSIONS

A first version of APSIM Next Gen. was released in October 2014. Much of the early work was on the infrastructure and user interface. A light weight kernel with a simple inter-model API has been built that runs

approximately seven times quicker than APSIM 7.7. With more profiling, it is expected that this will further improve. This opens up new opportunities for modellers and enables new scenarios to be examined.

There have been challenges along the way though. Using MONO to build cross platform software has been a challenge, particularly on OSX. We have gradually leaned what works well and what does not. There are clearly bugs in MONO that make positioning items on the user interface and relying on event order problematic on different operating systems. There have been some successes though. The APSIM next gen. models and user interface work well on LINUX (partially on OSX) and a proof-of-concept build showed that the models (not the user interface) run well on Android smart phones and tablets (using Xamarin - http://xamarin.com). It is expected that .NET compatibility will improve, particularly now that Microsoft is contributing source code to the MONO library.

We currently have a range of plant models released or under development. We are finding the new software development process outlined in the previous section provides a quicker 'time to release' for new models, with better, more up-to-date documentation and better testing.

Developing the new GitHub workflow and getting model developers who aren't software developers to use it, has been a challenge. It is very easy to commit files unintentionally and generally make a mess if the committer doesn't have a good mental model of how GIT and the workflow surrounding it works. While it is still a work in progress, we are trying to keep the committing process as simple as possible. Perhaps we allow model builders who are not comfortable with GIT workflow processes to use other mechanisms. For example, sharing sites like DropBox offer the ability to share files and manage changes between users. Using this approach they would submit their changes to the software team when they were ready. While this approach doesn't offer versioning and the ability to see and compare revisions, it does offer a practical workflow for some model developers.

One thing we have noticed over the years is the advantage of short iterations for development. By keeping development iterations short and committing regularly, developers can significantly lessen the pain of submitting fixes or new models into an APSIM release. When there is a long time period (> a week!) between beginning development of a new feature or defect fix and merging changes back into the repository, a significant amount of time will be spent on merge conflicts where another developer has changed the same file. Shortening the gap between development iterations, shortens the amount of time spent merging changes.

The fundamental philosophy behind the development of APSIM Next Gen. is twofold. Simplicity is paramount. We continually ask ourselves, "What is the minimum amount of code to make this work?" and "What is the most intuitive way to do this?" and "How would the user expect to do this?". Trying to keep it simple is anything but simple! In the past we have produced over engineered solutions that weren't needed and have provided multiple ways of doing things. We are trying not to make the same mistakes this time. The second principle relates to 'cruft' removal. Twenty four years of evolutionary development has left APSIM 7.7 with many band aids, workarounds and patches for fixes. We are attempting to reengineer an infrastructure and a suite of models that don't have these. The question is how long will our pristine, clean infrastructure stay in this state?

It has been a lot of fun 'starting from scratch'. The design decisions we have made, have been informed from our experiences with APSIM development over a long time period. We haven't thrown away 24 years of development, rather we have thrown out the bits we don't need any more and built upon and improved the bits we do need. APSIM 7.7 will continue to be released for several more years while we transition to APSIM Next Gen. The hope is that in three to five years, APSIM Next Generation will have the required functionality to satisfy the majority of users.

REFERENCES

Bell LW, Hargreaves JNG, Lawes RA, Robertson MJ (2009) Sacrificial grazing of wheat crops: identifying tactics and opportunities in Western Australia's grainbelt using simulation approaches. *Animal Production Science* **49**, 797–806. doi:10.1071/AN09014.

Brown HE, Huth N, Holzworth D (2011) A potato model built using the APSIM Plant.NET Framework. InChan F, Marinova D, Anderssen RS (eds) 961–967 ://WOS:000314989300130.

Brown HE, Huth NI, Holzworth DP, Teixeira EI, Zyskowski RF, Hargreaves JNG, Moot DJ (2014) Plant Modelling Framework: Software for building and running crop models on the APSIM platform. *Environmental Modelling & Software* **62**, 385–398. doi:10.1016/j.envsoft.2014.09.005.

Carberry PS, Hochman Z, McCown RL, Dalgliesh NP, Foale MA, Poulton PL, Hargreaves JNG, Hargreaves DMG, Cawthray S, Hillcoat N, Robertson MJ (2002) The FARMSCAPE approach to decision support: farmers', advisers', researchers' monitoring, simulation, communication and performance evaluation. *Agricultural Systems* **74**, 141–177. doi:10.1016/s0308-521x(02)00025-2.

Carberry PS, Liang W-L, Twomlow S, Holzworth DP, Dimes JP, McClelland T, Huth NI, Chen F, Hochman Z, Keating BA (2013) Scope for improved eco-efficiency varies among diverse cropping systems. *Proceedings of the National Academy of Sciences of the United States of America* **110**, 8381–6. doi:10.1073/pnas.1208050110.

Chapman S, Cooper M, Podlich DW, Hammer G (2003) Evaluating plant breeding strategies by simulating gene action and dryland environment effects. *Agron J* **95**, 99–113.

Hammer GL, van Oosterom E, McLean G, Chapman SC, Broad I, Harland P, Muchow RC (2010) Adapting APSIM to model the physiology and genetics of complex adaptive traits in field crops. *Journal of Experimental Botany* **61**, 2185–2202. doi:10.1093/jxb/erq095.

Hochman Z, Gobbett D, Holzworth D, McClelland T, van Rees H, Marinoni O, Garcia JN, Horan H (2012) Quantifying yield gaps in rainfed cropping systems: A case study of wheat in Australia. *Field Crops Research* **136**, 85–96. doi:10.1016/j.fcr.2012.07.008.

Hochman Z, Holzworth DP, Hunt JR (2009) Potential to improve on-farm wheat yields and WUE in Australia. *Crop and Pasture Science* **60**, 708–716.

Hochman Z, van Rees H, Carberry PS, Hunt JR, McCown RL, Gartmann A, Holzworth D, van Rees S, Dalgliesh NP, Long W, Peake AS, Poulton PL, McClelland T (2009) Re-inventing model-based decision support with Australian dryland farmers. 4. Yield Prophet® helps farmers monitor and manage crops in a variable climate. *Crop and Pasture Science* **60**, 1057–1070.

Holzworth DP, Huth NI, deVoil PG (2011) Simple software processes and tests improve the reliability and usefulness of a model. *Environmental Modelling & Software* **26**, 510–516.

Holzworth DP, Huth NI, deVoil PG, Zurcher EJ, Herrmann NI, McLean G, Chenu K, van Oosterom EJ, Snow V, Murphy C, Moore AD, Brown H, Whish JPM, Verrall S, Fainges J, Bell LW, Peake AS, Poulton PL, Hochman Z, Thorburn PJ, Gaydon DS, Dalgliesh NP, Rodriguez D, Cox H, Chapman S, Doherty A, Teixeira E, Sharp J, Cichota R, Vogeler I, Li FY, Wang E, Hammer GL, Robertson MJ, Dimes JP, Whitbread AM, Hunt J, van Rees H, McClelland T, Carberry PS, Hargreaves JNG, MacLeod N, McDonald C, Harsdorf J, Wedgwood S, Keating BA (2014) APSIM – Evolution towards a new generation of agricultural systems simulation. *Environmental Modelling & Software* **62**, 327–350. doi:10.1016/j.envsoft.2014.07.009.

Holzworth DP, Huth NI, de Voil PG (2010) Simplifying environmental model reuse. *Environmental Modelling and Software* **25**, 269–275.

Holzworth DP, Snow V, Janssen S, Athanasiadis IN, Donatelli M, Hoogenboom G, White JW, Thorburn P (in press) Agricultural production systems modelling and software: Current status and future prospects. *Environmental Modelling & Software*. doi:10.1016/j.envsoft.2014.12.013.

Huth NI, Banabas M, Nelson PN, Webb M (2014) Development of an oil palm cropping systems model: Lessons learned and future directions. *Environmental Modelling & Software* **62**, 411–419. doi:10.1016/j.envsoft.2014.06.021.

Huth NI, Carberry PS, Poulton PL, Brennan LE, Keating BA (2002) A framework for simulating agroforestry options for the low rainfall areas of Australia using APSIM. *European Journal of Agronomy* **18**, 171–185.

Huth NI, Henderson C, Peake A (2009) Development and testing of a horticultural crop model within APSIM. InAnderssen RS, Braddock RD, Newham LTH (eds) 526–532 ://WOS:000290045000077.

Jeffries R, Anderson A, Hendrickson C (2001) 'Extreme Programming Installed.' (Copyright (c) Addison-Wesley)

Lilley JM, Moore AD (2009) Trade-offs between productivity and ground cover in mixed farming systems in the Murrumbidgee catchment of New South Wales. *Animal Production Science* **49**, 837–851. doi:10.1071/AN09011.

Luo Z, Wang E, Baldock J, Xing H (2014) Potential soil organic carbon stock and its uncertainty under various cropping systems in Australian cropland. *Soil Research* **52**, 463–475. doi:10.1071/SR13294.

Luo Z, Wang E, Sun OJ, Smith CJ, Probert ME (2011) Modeling long-term soil carbon dynamics and sequestration potential in semi-arid agro-ecosystems. *Agricultural and Forest Meteorology* **151**, 1529–1544. doi:10.1016/j.agrformet.2011.06.011.

McCown RL, Carberry PS, Hochman Z, Dalgliesh NP, Foale MA (2009) Re-inventing model-based decision support with Australian dryland farmers. 1. Changing intervention concepts during 17 years of action research. *Crop and Pasture Science* **60**, 1017–1030.

Messina CD, Podlich D, Dong Z, Samples M, Cooper M (2011) Yield-trait performance landscapes: from theory to application in breeding maize for drought tolerance. *Journal of Experimental Botany* **62**, 855–868. doi:10.1093/jxb/erq329.

Moore AD, Bell LW, Revell DK (2009) Feed gaps in mixed-farming systems: insights from the Grain & Graze program. *Animal Production Science* **49**, 736–748. doi:10.1071/AN09010.

van Rees H, McClelland T, Hochman Z, Carberry P, Hunt J, Huth N, Holzworth D (2014) Leading farmers in South East Australia have closed the exploitable wheat yield gap: Prospects for further improvement. *Field Crops Research* **164**, 1–11. doi:10.1016/j.fcr.2014.04.018.

Rodriguez D, Cox H, deVoil P, Power B (2014) A participatory whole farm modelling approach to understand impacts and increase preparedness to climate change in Australia. *Agricultural Systems* **126**, 50–61. doi:10.1016/j.agsy.2013.04.003.

Snow VO, Rotz CA, Moore AD, Martin-Clouaire R, Johnson IR, Hutchings NJ, Eckard RJ (2014) The challenges – and some solutions – to process-based modelling of grazed agricultural systems. *Environmental Modelling & Software* **62**, 420–436. doi:10.1016/j.envsoft.2014.03.009.

Teixeira E., Brown H., Fletcher AL, Hernandez-Ramirez G, Soltani A, Viljanen-Rollinson S, Horrocks A, Johnstone P (2012) Adapting broad acre farming to climate change. *In: Impacts of Climate Change on Land-based Sectors and Adaptation Options Clark, AJ; Nottage, RAC (eds) Technical Report to the Sustainable Land Management and Climate Change Adaptation Technical Working Group, Ministry for Primary Industries, 408 p*.

Thorburn PJ, Robertson MJ, Clothier BE, Snow VO, Charmley E, Sanderman J, Teixeira E, Dynes RA, Hall A, Brown H, Howden SM, Battaglia M (2012) 'Climate change and agriculture in Australia and New Zealand. In: Cynthia Rosenzweig and Daniel Hillel (Eds), IPC Series on Climate Change Impacts, Adaptation and Mitigation Volume 2, Handbook of Climate Change and Agroecosystems, Global and Regional Aspects and Implications.' (Imperial College Press, London)

Wang J, Wang E, Liu DL (2011) Modelling the impacts of climate change on wheat yield and field water balance over the Murray-Darling Basin in Australia. *Theoretical and Applied Climatology* **104**, 285–300. doi:10.1007/s00704-010-0343-2.

Whish JPM, Castor P, Carberry PS, Peake AS (2007) On-farm assessment of constraints to chickpea (Cicer Arietinum) production in marginal areas of northern Australia. *Experimental Agriculture* **43**,. doi:10.1017/S0014479707005297.