Estimación de parámetros frecuentista y bayesiana en modelos dinámicos de crecimiento de cultivos en invernadero y su programación en Matlab y R

LÓPEZ-CRUZ, Irineo Lorenzo, FITZ-RODRÍGUEZ, Efrén, SALAZAR-MORENO, Raquel, RUIZ-GARCÍA, Agustín y ROJANO-AGUILAR, Abraham

I. López, E. Fitz, R, Salazar, A. Ruiz y A. Rojano

Postgrado en Ingeniería Agrícola y Uso Integral del Agua, Universidad Autónoma Chapingo, Chapingo Estado de México
ilopez@correo.chapingo.mx

F. Pérez, E. Figueroa, L. Godínez, R. García (eds.) Ciencias de la Economía y Agronomía. Handbook T-II.-©ECORFAN, Texcoco de Mora, México, 2017.

Abstract

Matlab is a programming environment used in teaching and researching for dynamic simulation of systems. R is a programming language originally used to data Statistical Analysis, but nowadays also it is applied to simulation of dynamical systems. In the current work a comparison of a parameter estimation by classical and Bayesian methods and their programming in Matlab and R language is carried out. A lettuce crop growth dynamic model was programmed first. Then the dynamic model was solved using the numerical methods available in both environments. Next, two models parameters were estimated using the nonlinear least squares method accessible in Matlab and R. Finally, the Bayesian method known as Monte Carlo Markov Chains was programmed. Differences and similarities between advantages and disadvantages of both programming environments are emphasized.

2 Introducción

La estimación de parámetros de un modelo matemático dinámico consiste en la determinación de valores para los parámetros inciertos incorporando mediciones de algunas variables de salida del modelo, de tal forma que las predicciones del modelo se ajusten lo más posible a las observaciones (Wallach, 2011). Este proceso también conocido como calibración es una etapa crucial y necesaria en el desarrollo del modelo dinámico de un sistema ya que podría mejorar la calidad predictiva del modelo matemático. El problema de estimación de estimación de parámetros (Ioslovich et al., 2002) consiste en encontrar el conjunto de valores para los parámetros (θ) del modelo dinámico, de tal forma que minimice un criterio ($V(\theta)$). Normalmente el cuadrado medio del error entre mediciones obtenidas del sistema real ($z(k)$) y las predicciones del modelo ($y(k \mid \theta)$).

$$V(\theta) = e^T(k)e(k) = \sum_{k=1}^{N} \left(z(k) - y(k \mid \theta)\right)^2 \tag{2}$$

La estimación de parámetros en un modelo implica la realización de experimentos para poder disponer de mediciones del sistema real. Y posteriormente el planteamiento y solución de un problema de optimización. En general, el problema de estimación de parámetros en modelos dinámicos puede abordarse mediante dos enfoques: el frecuentista y el Bayesiano (Makowski et al., 2006). Hasta ahora el primer enfoque es más común en ambientes de agricultura controlada. El primer enfoque usa mínimos cuadrados o máxima verosimilitud y el segundo el teorema de Bayes. En el paradigma clásico los parámetros son estimados solamente usando las mediciones y no son considerados como variables aleatorias. En contraste, en el paradigma Bayesiano, los parámetros del modelo son considerados explícitamente como variables aleatorias y tanto el conocimiento a priori como los datos son usados para calcular una distribución a posteriori. Los métodos basados en este paradigma son los denominados muestreos basados en importancia como la Estimación de Incertidumbre con Verosimilud Generalizada (GLUE por sus siglas en inglés) y las Cadenas de Markov Monte Carlo (MCMC) como el método de Metropolis-Hastings (Makowski et al., 2002; Wallach et al, 2014). Tanto el paradigma clásico de estimación de parámetros como el Bayesiano son motivo de investigación en la actualidad ya que existen varias preguntas abiertas tanto teóricas como prácticas por ser respondidas (Wallach et al., 2014). Aunque el procedimiento de estimación de parámetros basado el métodos clásicos está bien y mejor establecido es también bien conocido que estos métodos permiten estimar con precisión un limitado número de parámetros (menos de 10). Por su parte, aunque el enfoque Bayesiano es atractivo un procedimiento bien establecido no existe todavía debido a su alto costo computacional. A pesar de sus ventajas aparentes los métodos Bayesianos se ha usado poco para estimar parámetros en modelos dinámicos para crecimiento de cultivos (Makowski et al., 2002; Iisumi et al., 2009; Ceglar et al., 2011; Dzotsi, et al., 2015).

Matlab (The MathWorks, Inc, 2012) es un ambiente de programación comercial ampliamente usado en enseñanza e investigación. Su gran atractivo es que presenta herramientas altamente especializadas y confiables para resolver problemas, tales como Simulink y el ODE suite, para simulación dinámica y el Optimization Toolbox para resolver problemas de optimización. R es un entorno de programación libre para cómputo estadístico y gráfico (https://www.r-project.org/). Sin embargo, cada día tiene disponibles nuevas librerías disponibles incluyendo aquellas para simular modelos dinámicos y resolver problemas de optimización. Recientemente R ha sido usado para simular y analizar modelos de crecimiento de cultivos (Wallach et al., 2014). Por lo anterior, en el presente trabajo se presenta una comparación de la programación en Matlab-Simulink y R de la estimación de parámetros clásica y Bayesiana de un modelo dinámico para un cultivo de lechugas.

2.1 Materiales y métodos

2.1.1 El modelo dinámico para crecimiento de lechugas bajo invernadero y su programación en Matlab y R

Un modelo dinámico de dos estados, para crecimiento de lechugas, fue propuesto en el pasado (van Henten, 1994; van Henten and van Straten, 1994). El primer estado es el peso seco estructural X_s (gm^{-2}) como la que constituye paredes celulares y citoplasma. La segunda variable de estado es la biomasa no estructural X_{ns} (gm^{-2}) cuyos principales componentes son la glucosa, sucrosa y almidón. La tasa de cambio de las variables predichas por el modelo está dada por dos ecuaciones diferenciales no lineales:

$$\frac{dX_{ns}}{dt} = c_\alpha f_{phot} - r_{gr} X_s - f_{resp} - \frac{1-c_\beta}{c_\beta} X_s \tag{2.1}$$

$$\frac{dX_s}{dt} = r_{gr} X_s \tag{2.2}$$

Donde f_{phot}($gm^{-2}s^{-1}$) es la fotosíntesis bruta, f_{resp} ($gm^{-2}s^{-1}$) es la respiración de mantenimiento r_{gr} (s^{-1}) es la tasa de crecimiento específica y c_α y c_β (adimensional) son parámetros. La descripción completa del modelo puede encontrarse en otros artículos (van Henten, 1994; van Henten and van Straten, 1994). La descripción de los parámetros del modelo se presenta en la Tabla 2. Las variables de entrada del modelo (variables conductoras) son la radiación fotosintéticamente activa (Wm^{-2}), la temperatura del aire (°C) y la concentración de dióxido de carbono (ppm).

Tabla 2 Parámetros del modelo para lechugas de dos variables de estado de van Henten

Descripción	Valor	Unidades	Referencia
Factor de conversión de CO_2 asimilado a azucares (c_α)	0.68	-	Calculado
Factor de rendimiento (c_β)	0.80	-	Sweeny et al., 1981
Tasa de crecimiento a saturación ($c_{gr,\max}$)	5.00×10^{-6}	(s^{-1})	Sweeny et al., 1981
Factor de crecimiento Q_{10} ($c_{Q10,gr}$)	1.60	-	Sweeny et al., 1981
Coeficiente de respiración de mantenimiento de brotes a 25°C ($c_{resp,sht}$)	4.47×10^{-7}	(s^{-1})	van Keulen et al., 1982

Coeficiente de respiración de mantenimiento de las raíces a 25°C ($c_{resp,sht}$)	1.16×10^{-7}	(s^{-1})	van Keulen et al., 1982
Factor Q_{10} de tasa de respiración de mantenimiento ($c_{Q10,resp}$)	2.00	-	van Keulen et al., 1982
Razón de biomasa de raíces y biomasa total (c_{μ})	0.15	-	Medición
Coeficiente de extinción de la radiación (c_k)	0.90	-	Goudriaan and van Laar, 1994
Relación de area foliar (c_{lar})	75.00×10^{-3}	m^2g^{-1}	van Henten, 1994
Eficiencia de uso de la luz a altos niveles (c_{ε})	17.00×10^{-6}	gJ^{-1}	Goudriaan et al., 1985
Efecto de la temperatura sobre la conductancia de carboxilación ($c_{car,1}$)	-1.32×10^{-5}	$ms^{-1}C^{-2}$	van Henten, 1994
Efecto de la temperatura sobre la conductancia de carboxilación ($c_{car,2}$)	5.94×10^{-4}	$ms^{-1}C^{-2}$	van Henten, 1994
Efecto de la temperatura sobre la conductancia de carboxilación ($c_{car,3}$)	-2.64×10^{-3}	ms^{-1}	van Henten, 1994
Conductancia estomática (c_{stm})	0.007	ms^{-1}	Stanghellini, 1987
Conductancia de la capa límite (c_{bnd})	0.004	ms^{-1}	Stanghellini, 1987
Punto de compensación de CO_2 a 20°C (c_{Γ})	7.32×10^{-5}	gm^{-3}	Goudriaan et al., 1985
Factor Q_{10} del punto de compensación de CO_2 ($c_{Q10,\Gamma}$)	2.00	-	Goudriaan et al., 1985
Densidad del dióxido de carbono (c_w)	1.83×10^{-3}	gm^{-3}	Constante física

El modelo matemático fue programado en el ambiente Matlab-Simulink mediante una subrutina compilada en lenguaje C con la finalidad de reducir el tiempo de simulación. Para llevar a cabo la integración numérica se usó el método de Dormand-Prince (función ode45.m de Matlab) de cuarto orden con tamaño de paso variable, una tolerancia relativa de 1.0×10^{-12} y una tolerancia absoluta de 1.0×10^{-14}. Para todas las simulaciones y estimación de parámetros se usaron los datos de un experimento llevado a cabo en Chapingo, México durante 2001 (Ramírez et al., 2001). En el caso de lenguaje R, el modelo dinámico fue programado como una subrutina. Para la integración numérica se usó el General Solver for Ordinary Differential Equations de la biblioteca *deSolve*. Se usó el método de integración de Dormand-Prince contenido en la función "ode45", con las mismas precisiones relativas y absolutas especificadas en el caso de Matlab. Se verificó que los dos programas produjeran los mismos valores de las variables de estado. De esta forma se comprobó que en ambos casos el modelo matemático y el modelo computacional eran los mismos.

2.1.2 Estimación frecuentista de parámetros y su programación en Matlab y R

En un primer momento fueron estimados solo dos de los parámetros para los que el modelo es más sensible (van Henten and van Straten 2004). Así, el factor de rendimiento (c_{β}) y la razón de área foliar (c_{lar}) fueron calibrados. Dado el vector de parámetros $\hat{\boldsymbol{\theta}} = \lfloor c_{lar} \quad c_{\beta} \rfloor$ a ser estimados. Mediante mínimos cuadrados no lineales el siguiente problema de optimización fué planteado y resuelto:

$$\text{argmin} V(\boldsymbol{\theta}) \tag{2.3}$$

Donde $\text{argmin} V(\boldsymbol{\theta})$ significa el argumento de la función V que la minimiza.

$$V(\boldsymbol{\theta}) = \sum_{k=1}^{N} \left[z(k) - y(k \mid \boldsymbol{\theta}) \right]^2 = \sum_{k=1}^{N} e(k \mid \boldsymbol{\theta})^2 \tag{2.4}$$

Donde $z(k)$ y $y(k \mid \boldsymbol{\theta})$ representan el peso seco de un cultivo de lechugas medido y predicho, respectivamente, en los instantes discretos k. Con la rutina mínimos cuadrados no lineales (lsqnonlin.m) que forma parte del Optimization Toolbox de Matlab es possible calcular no solamente los parámetros sino también su asociada matriz Jacobiana matrix (Φ) y los residuales ($e(k \mid \boldsymbol{\theta})$). La matriz de covarianzas $\text{cov}(\hat{\boldsymbol{\theta}})$ correspondiente a los parámetros estimados puede ser calculada como sigue:

$$\text{cov}(\hat{\boldsymbol{\theta}}) = \sigma_{\varepsilon}^2 \left(\Phi^T \Phi \right)^{-1} \tag{2.5}$$

Con la varianza σ_{ε}^2 del error de medición ε estimada mediante:

$$s^2 = \frac{V(\hat{\boldsymbol{\theta}})}{N - N_p} \tag{2.6}$$

Donde N es el número de datos (mediciones) y N_p es el número de parámetros estimados. A partir de estos cálculos se puede tener una idea del nivel de precisión con que se estiman los parámetros.

El problema anterior fue resuelto en Matlab usando la función *lsqnonlin*, la cual permute resolver problemas de mínimos cuadrados. Para esto es necesario escribir una subrutina que devuelve un vector diferencia entre las predicciones y mediciones. Solo se usó la diferencia entre las predicciones y mediciones de la variable biomasa total. Pero es posible usar otras variables medidas. Se especificó el mismo intervalo de incertidumbre tanto en Matlab como en R como sigue: $0.001 \leq c_{lar} \leq 0.1$ y $0.4 \leq c_{\beta} \leq 0.9$. En el caso de lenguaje R se usó el procedimiento Nonlinear Least Squares que se encuentra en la función *nls*. En forma similar a Matlab para usar la función *nls* en R se requiere programar una función que devuelva el valor de la variable predicha correspondiente a cada iteración con los valores probados de los parámetros que están siendo estimados. Tanto en Matlab como en R estás subrutinas llevan a cabo una integración numérica (simulación) del modelo dinámico en cada iteración de proceso de búsqueda. La función *lsqnonlin* de Matlab permite calcular no solo el vector de parámetros estimados sino también su residuales y matriz Jacobiana correspondientes. La función *nls* de R permite calcular los coeficientes estimados, varias estadísticas asociadas a la estimación así como el comportamiento del modelo ajustado. La función *nls* de R es mucho más completa que *lsqnonlin* de Matlab.

2.1.3 Estimación Bayesiana de parámetros y su programación en Matlab y R

El enfoque Bayesiano para estimación de parámetros consta de dos etapas (Wallach et al., 2014). Primeramente se requiere definir una distribución de parámetros *a priori* cuya densidad se denota como $P(\theta)$. Esta resume el estado del conocimiento sobre los valores de los parámetros. Generalmente se usa una distribución uniforme o Gausiana. También se necesita especificar una distribución de probabilidades de los errores del modelo. El segundo paso consiste en calcular una distribución de probabilidades nueva llamada distribución *a posterior*i cuya densidad es $P(\theta \mid Y)$. Esta permite actualizar nuestra creencia en los valores de los parámetros θ incorporando las mediciones Y. La distribución a posteriori es calculada usando el teorema de Bayes.

$$P(\theta|Y)=\frac{P(Y|\theta)P(\theta)}{P(Y)} \tag{2.7}$$

Donde $P(Y|\theta)$ es una función de verosimilitud, la cual proporciona la probabilidad de las mediciones para valores dados de los parámetros del modelo. $P(Y)$ es la distribución de las observaciones independientemente de los parámetros. Desafortunadamente, la complejidad de los modelos dinámicos de cultivos hace que $P(Y)$ no pueda ser determinada, por lo tanto se requiere del uso de algoritmos numéricos para aproximar la distribución a posteriori. La principal idea es usar métodos iterativos para generar una muestra grande de valores de los parámetros estimados cuya distribución de probabilidades aproxime la distribución a posteriori. Actualmente existen dos métodos genéricos (Wallach et al., 2014) conocidos como muestreo de importancia y el procedimiento Cadenas de Markov Monte Carlo (MCMC). Un ejemplo de los primeros métodos es el análisis de incertidumbre Bayesiano conocido como Generalized Likelihood Uncertainty Estimation (GLUE). En el caso de MCMC el algoritmo de Metropolis-Hastings es conocido desde hace algún tiempo (Gelman et al., 2014;). En el presente trabajo se programó tanto en Matlab como en lenguaje R el algoritmo de Metropolis-Hastings tal como se describe en Wallach et al (2104).

Primeramente, se abordó el problema de estimar simultáneamente dos parámetros; el cociente de área foliar (c_{lar}) y la desviación estándar de los residuales del error del modelo (σ). La distribución anterior de c_{lar} se definió como una distribución normal con media 0.02 y desviación estándar 0.01. La distribución anterior de σ fue definida como una distribución uniforme con limites inferior y superior como 0 y 10. Después de varias pruebas, las distribuciones correspondientes para las Cadenas de Markov Monte Carlo fueron especificadas como $c_{lar}(i)\sim N(c_{lar}(i-1),0.02^2)$ y $\log(\sigma(i))\sim N(\log(\sigma(i-1)),0.8^2)$, respectivamente. Se generaron cadenas de 10000 iteraciones que necesitaron 3.5 horas de tiempo de computación en el caso de Matlab y aproximadamente 8 horas en el caso del lenguaje R. Se usó una computadora con procesador Intel Core i7-4500U CPU@1.80GHz 2.40 GHz con sistema operativo Windows.

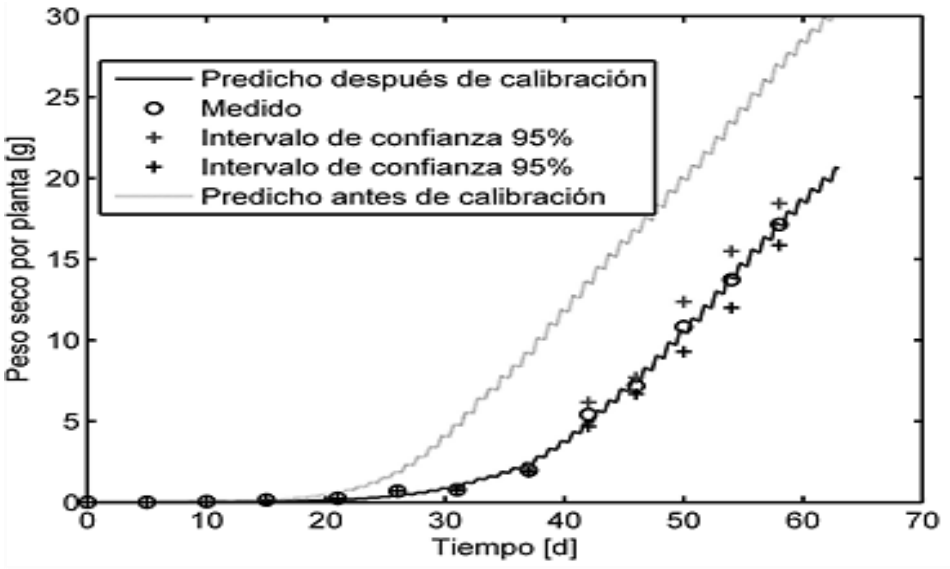
2.2 Resultados

2.2.1 Estimación de parámetros frecuentista

Los parámetros estimados en el caso de Matlab fueron $\hat{\theta}=[0.0471\ \ 0.6712]$ con una matriz de covarianzas $\text{cov}(\hat{\theta})=\begin{bmatrix}0.0000021943 & -0.000029986\\ -0.000029986 & 0.00043253\end{bmatrix}$. Esto arroja una desviación estándar para c_{lar} de 0.0015 (mg^{-1}) y un coeficiente de variación de 3.1%. En el caso de c_{β} estos valores fueron 0.0208 y 3.0%, respectivamente. En caso de R los parámetros estimados fueron $\hat{\theta}=[0.0471\ \ 0.6718]$ y las desviaciones estándar $STD_{C_{\beta}}=0.0206$ y $STD_{C_{lar}}=0.0013$. Las diferencias fueron pequeñas tanto en los valores estimados como en la precisión de la estimación. R no devuelve la matriz de covarianzas asociada a los parámetros, pero si las estadísticas coeficiente de correlación, desviaciones estándar y la significancia estadística de los parámetros. En este caso se encontró un alto valor de correlación para el parámetro c_{lar} (-0.97) debido a que este aparece en la función de intercepción de la radiación solar $1-\exp(-c_k\times c_{lar}(1-c_{\mu})x_s)$ donde se aprecia la interacción del parámetro relación de área foliar (c_{lar}) con el coeficiente de extinción de la radiación (c_k).

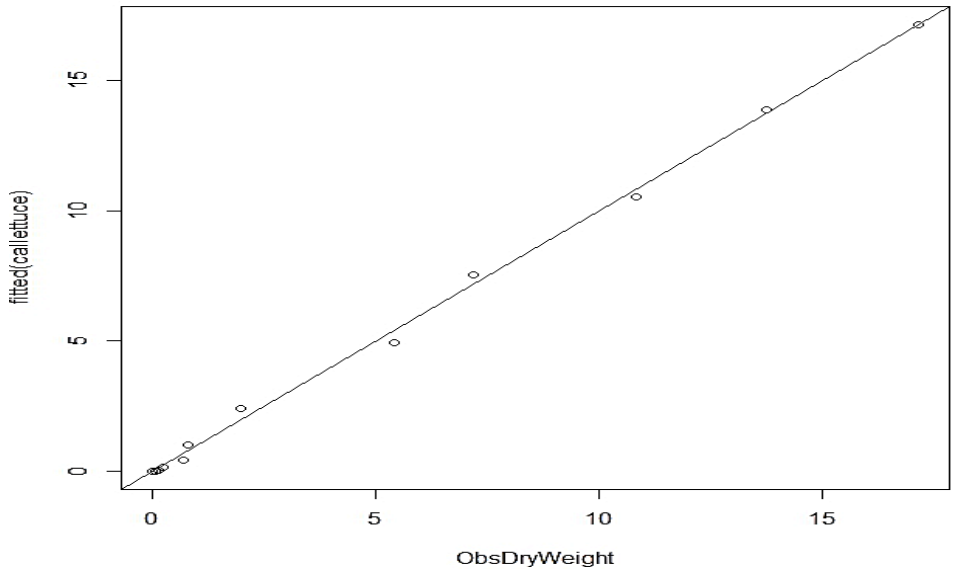
El Gráfico 2 muestra el comportamiento del modelo antes y después de la calibración de los dos parámetros. Después de la calibración el modelo sigue el comportamiento de las mediciones pues sus predicciones están dentro de intervalo de confianza del 95%.

Gráfico 2 Comportamiento del modelo antes y después de la estimación de dos de sus parámetros



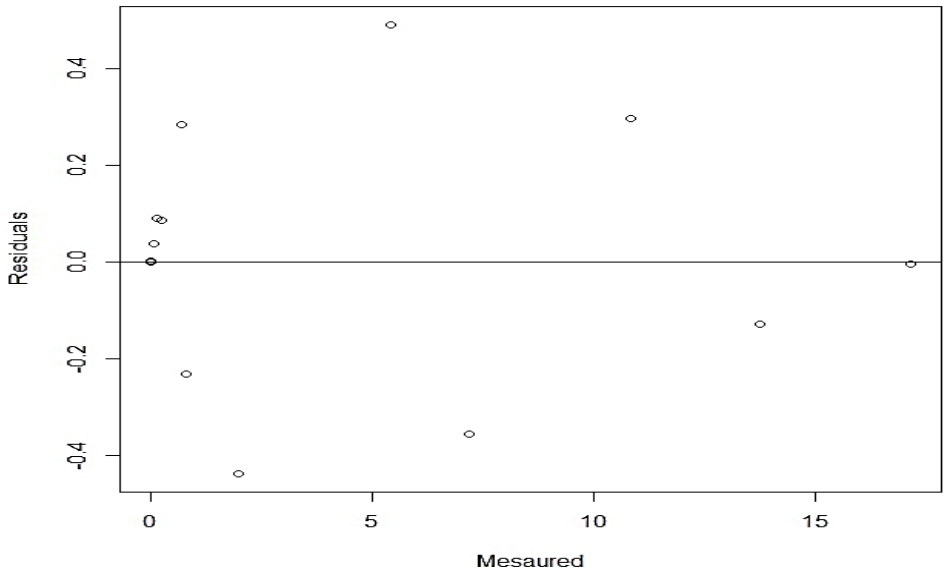
El Gráfico 2.1 muestra el ajuste del modelo, después de la calibración, usando la función *fitted* disponible en lenguaje R.

Gráfico 2.1 Biomasa total medida y estimada por el modelo calibrado



El Gráfico 2.2 muestra el comportamiento de las mediciones contra los residuales del modelo calculados con la función residuals disponible en R.

Gráfico 2.2 Comportamiento de los residuales comparados con las mediciones de la biomasa total



2.2.2 Estimación de parámetros Bayesiana

Gráfico 2.3 Resultados de 200 (A, C) y de 10000 (B, D) del algoritmo de Metropolis-Hastings programado en Matlab

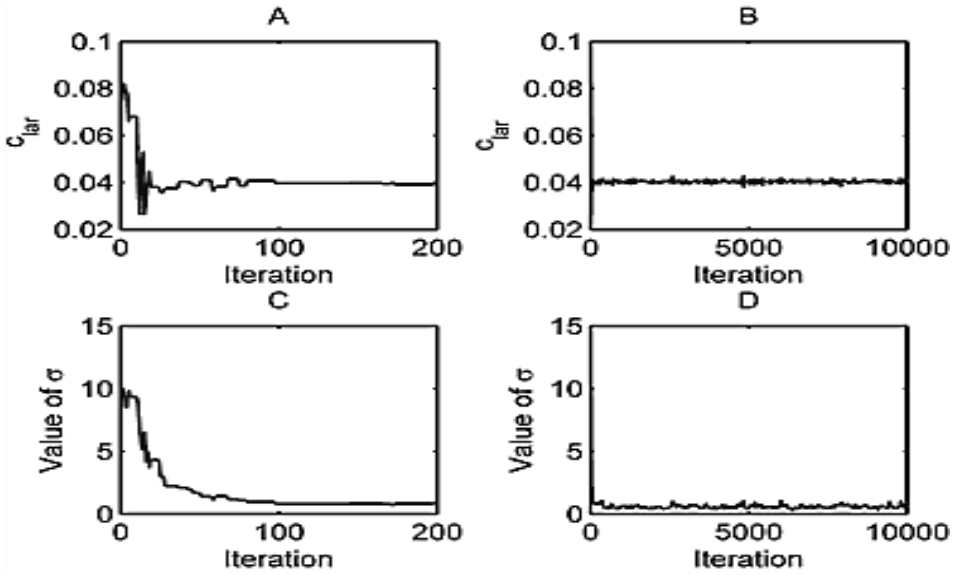
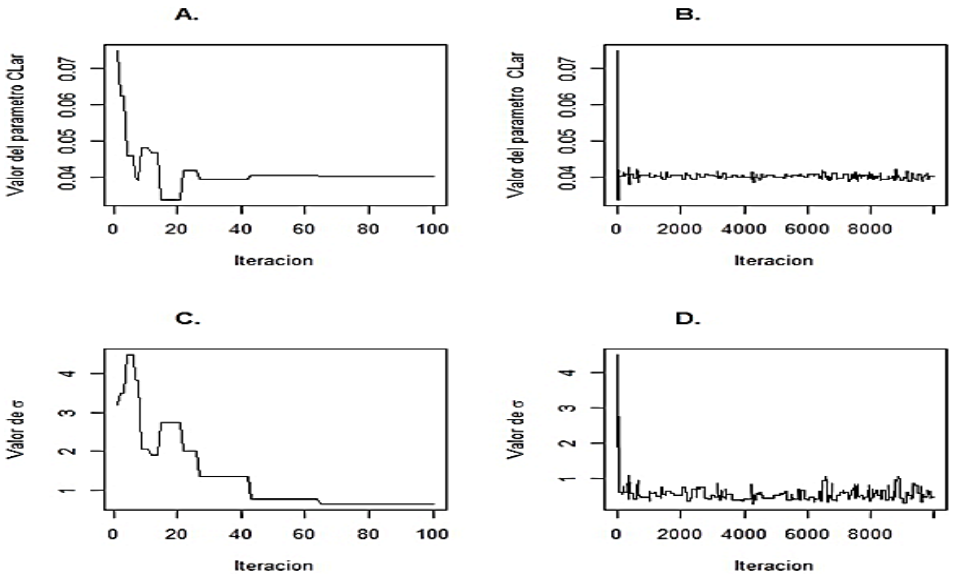


Gráfico 2.4 Resultados de 100 (A, C) y de 10000 (B, D) del algoritmo de Metropolis-Hastings programado en lenguaje R



El Gráfico 2.3 muestra el comportamiento de los parámetros estimados mediante el algoritmo Metropolis-Hastings, programado en Matlab, para diferentes iteraciones que indican la convergencia del algoritmo. El Gráfico 2.4 muestra el comportamiento del mismo algoritmo programado en lenguaje R. Puede observarse que después de 100 iteraciones el algoritmo empieza a oscilar alrededor de la media de ambos parámetros.

Gráfico 2.5 Auto correlaciones de los parámetros c_{lar} y de σ calculados por el algoritmo de Metropolis-Hastings programado en Matlab usando 10000 iteraciones

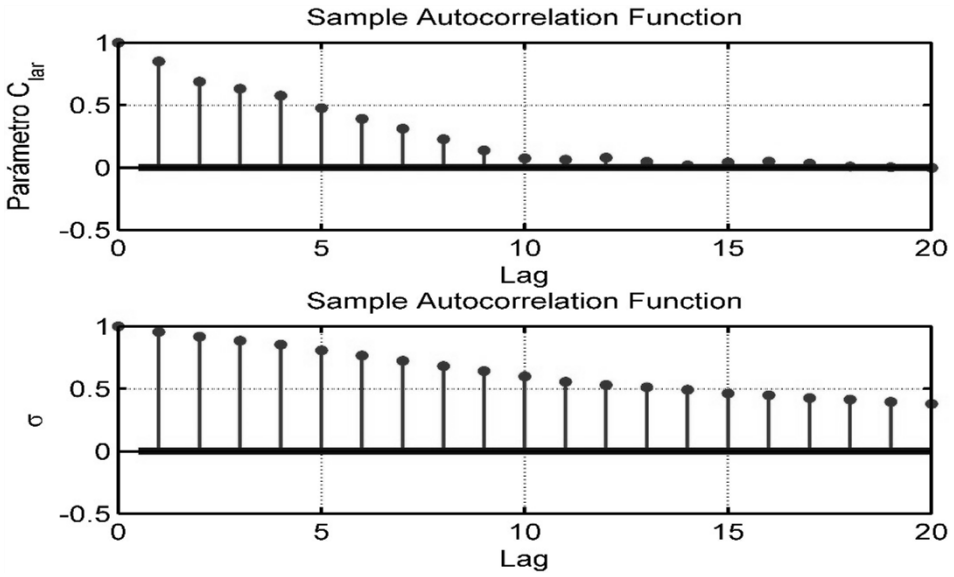
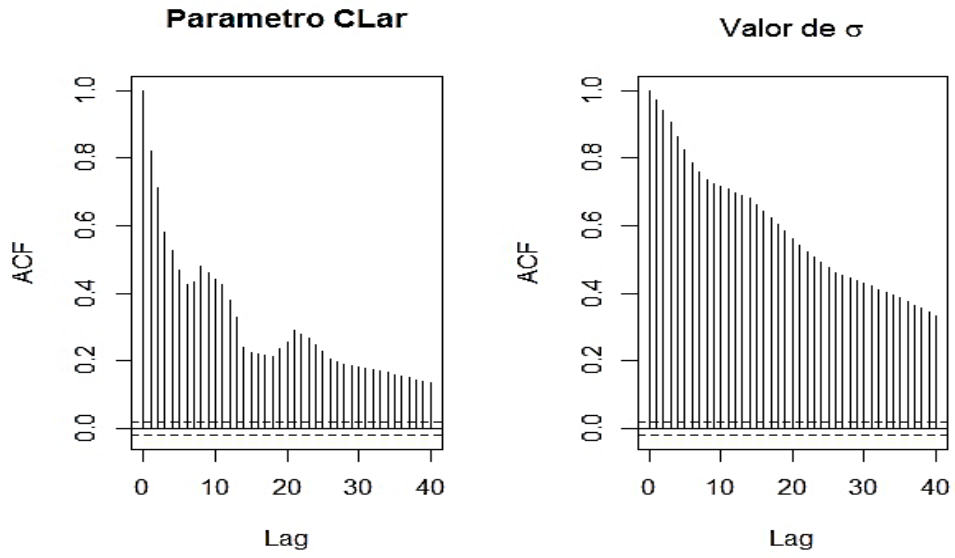


Gráfico 2.6 Auto correlaciones de los parámetros c_{lar} y de σ calculados por el algoritmo de Metropolis-Hastings programado en lenguaje R usando 10000 iteraciones

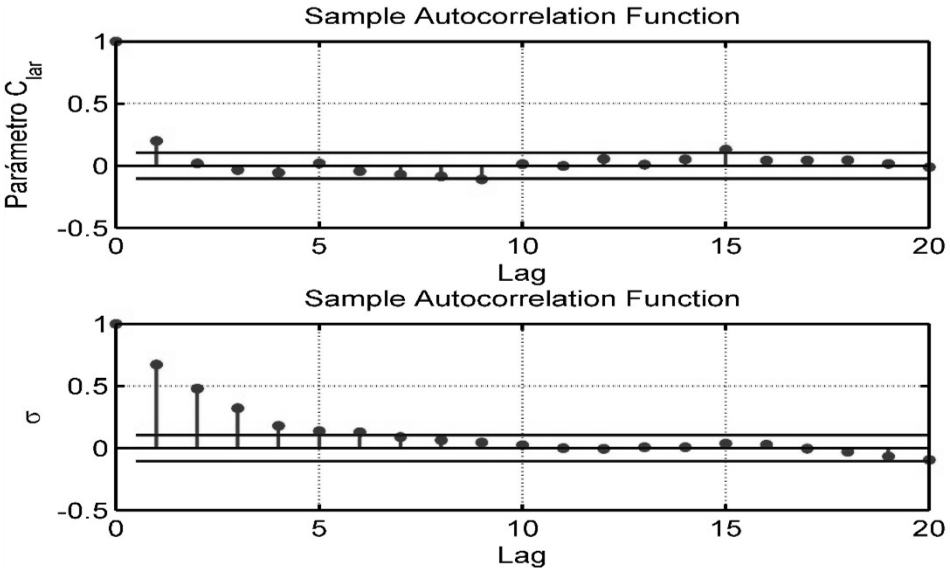


Es conocido que los valores de los parámetros estimados mediante el algoritmo Metropolis-Hastins estén generalmente altamente correlacionados debido a que la estimación del parámetro en una iteración actual depende de su valor en la iteración anterior (Gelman et al., 2014; Wallach et al., 2014). En nuestro caso al calcular las funciones de auto correlación se obtuvieron los resultados mostrados en el Gráfico 2.5 para el caso del algoritmo programado en Matlab y el Gráfico 2.6 para el caso del algoritmo programado en lenguaje R.

Las diferencias pueden deberse a los generadores de números pseudo-aleatorios y funciones generadoras de números pseudoaleatorios con una distribución normal (*randn* en Matlab y *rnorm* en R), funciones generadoras de números pseudoaleatorios con una distribución uniforme (*rand* en Matlab y *runif* en R), las funciones de densidad de probabilidades normal (*normpdf* en Matlab y *dnorm* en R) y uniforme (*unifpdf* en Matlab y *dunif* en R) usadas en ambos ambientes de programación. Pero es claro que las auto correlaciones deben ser eliminadas para no subestimar las varianzas posteriores (Wallach et al., 2014).

Para obtener series que sean casi sin correlaciones se considera un valor de una serie de 10 o más valores.

Gráfico 2.7 Auto correlaciones de los parámetros c_{lar} y de σ calculados por el algoritmo de Metropolis-Hastings programado en Matlab considerando uno de 20 valores generados por la MCMC



El Gráfico 2.7 muestra las auto-correlaciones de los parámetros estimados considerando uno de 20 valores de los últimos 7500 valores generados por el algoritmo Metropolis-Hastings programado en Matlab. Puede observarse que las auto correlaciones fueron prácticamente eliminadas especialmente en el caso del parámetro c_{lar}.

El Gráfico 2.8 muestra las auto-correlaciones de los parámetros estimados considerando uno de 20 valores de los últimos 7500 valores generados por el algoritmo Metropolis-Hastings programado en Lenguaje R. De manera similar a lo obtenido en Matlab, las auto correlaciones fueron prácticamente eliminadas.

Los Gráficos 2.9 y 2.10 muestran los histogramas correspondientes a 376 valores de los parámetros calculados por el algoritmo Metropolis-Hastings programado en Matlab y en lenguaje R, respectivamente.

Gráfico 2.8 Auto correlaciones de los parámetros c_{lar} y de σ calculados por el algoritmo de Metropolis-Hastings programado en lenguaje R, considerando uno de 20 valores generados por la MCMC

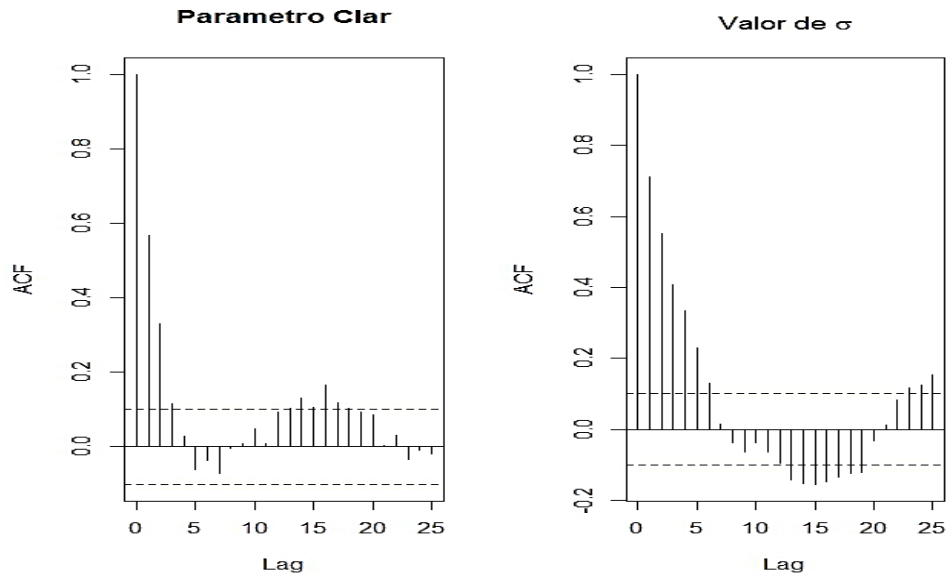


Gráfico 2.9 Histogramas de los parámetros c_{lar} y σ calculados por el algoritmo de Metropolis-Hastings programado en Matlab

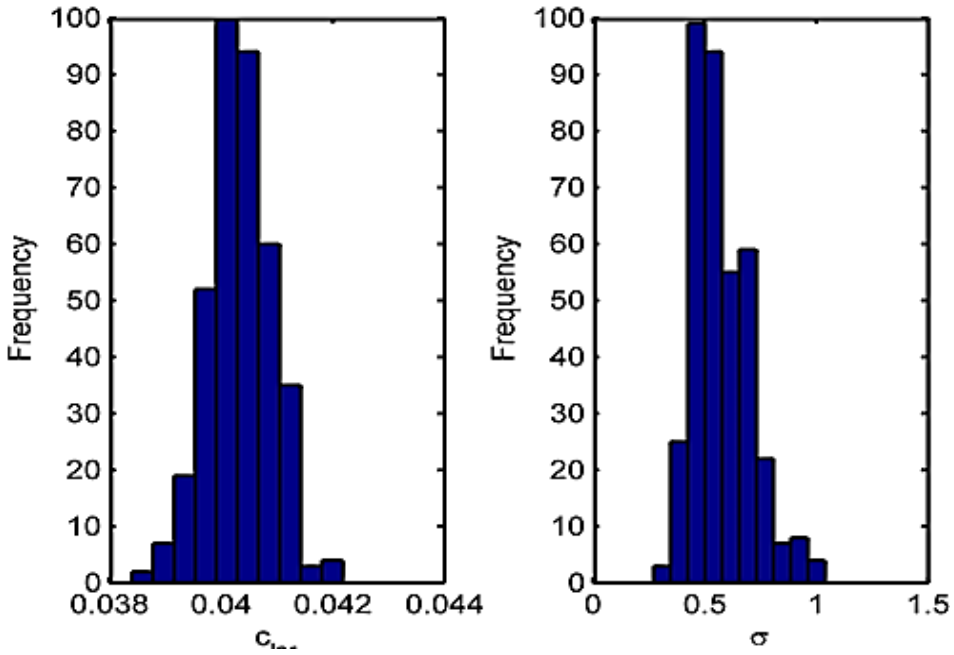
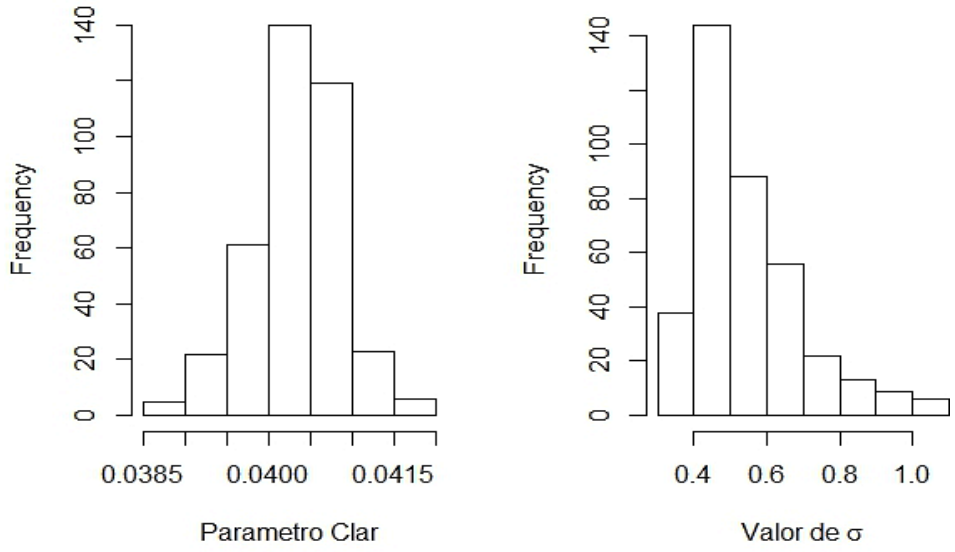


Gráfico 2.10 Histogramas de los parámetros c_{lar} y σ calculados por el algoritmo de Metropolis-Hastings programado en lenguaje R



En la Tabla 2.1 se presenta una comparación de los parámetros estimados mediante el algoritmo Metropolis-Hastings programado en Matlab y R. Se observan algunas diferencias, especialmente en la estimación del parámetro (σ) la desviación estándar de los residuales del error del modelo.

Tabla 2.1 Estadísticas asociadas a parámetros mediante el algoritmo de Metropolis-Hastings

Estadística	Programa en Matlab		Programa en R	
	c_{lar}	$\exp(\sigma)$	c_{lar}	$\exp(\sigma)$
Mínimo	0.0384	0.2710	0.0388	0.3157
Máximo	0.0422	1.0384	0.0418	1.0690
Media	0.0404	0.5756	0.0403	0.5467
Mediana	0.0403	0.5546	0.0403	0.5232
Desviación estándar	5.6783×10^{-4}	0.1303	5.4671×10^{-4}	0.1500
Coeficiente de variación	1.40%	22.63%	1.35%	27.44%

Las diferencias se pueden atribuir a las diferentes funciones generadoras de números pseudoaleatorios y funciones de densidad de probabilidades que se usaron para programar el algoritmo de Metropolis-Hastings en Matlab y en R. Sin embargo, en ambos casos se logró una estimación más precisa del parámetro relación de área foliar (c_{lar}) usando el método Bayesiano que con el método frecuentista. Con el método clásico de mínimos cuadrados el coeficiente de variación fue de 3.1% y con el método Bayesiano el coeficiente de variación fue de 1.40%. Esto muestra las ventajas potenciales del paradigma Bayesiano.

2.3 Discusión

2.3.1 Programación de modelos dinámicos

Tanto Matlab como R permiten resolver numéricamente sistemas de ecuaciones diferenciales ordinarias no lineales. En ambos ambientes están disponibles los métodos numéricos más usados. Sin embargo, Matlab permite programar los modelos dinámicos de manera más versátil: mediante subrutinas o funciones (*function*) de Matlab y usar directamente métodos de integración numérica, mediante la herramienta de programación de alto nivel Simulink y mediante Simulink combinado con subrutinas en lenguaje C (C-MEX s-sfunctions) compiladas. Esto último es especialmente útil en el caso de modelos de biosistemas que son complejos: contienen muchas variables de estado y son altamente no lineales, ya que permite reducir considerablemente el tiempo de computación.

Por el contrario, el lenguaje R solo permite programar el modelo matemático en una subrutina o función (*function*). Por supuesto que en Matlab se necesita mayor conocimiento de programación para poder usar todas sus ventajas potenciales.

En R es más simple la programación de la subrutina conteniendo el modelo matemático. En Matlab es muy simple e intuitivo programar el modelo matemático en el ambiente de simulación Simulink. El tiempo de computación requerido para simular un modelo dinámico puede ser muy muy costoso en R ya que no es posible compilar el modelo matemático pues esencialmente se interpreta el código.

El tiempo de computación puede ser considerablemente reducido en Matlab mediante la programación del modelo matemático dinámico (ecuaciones diferenciales ordinarias no lineales) en lenguaje C y el uso de la herramienta Matlab Executable (C-MEX s-functions).

2.3.2 Estimación de parámetros frecuentista

Matlab cuenta con la una función *lsqnonlin* para resolver problemas de mínimos cuadrados como parte del Optimization Toolbox. El lenguaje R la función *nls* determina estimadores de mínimos cuadrados no lineales de parámetros de un modelo no lineal.

En constraste el lenguaje R permite llevar a cabo análisis estadístico más completos de la estimación de parámetros frecuentista. En Matlab se requiere de mayor conocimiento de programación para determinar la precisión de la estimación.

2.3.3 Estimación de parámetros Bayesiana

Ni Matlab ni R cuentan todavía mcon algoritmos para estimación de parámetros Bayesiana. En ambos ambientes de programación se requiere programar los algoritmos deseados ya sean Muestreo de importancia o Cadenas de Markov Monte Carlo (MCMC). Por esto, el uso de funciones generadoras de números pseudoaleatorios con una distribución normal, funciones generadoras de números pseudoaleatorios con una distribución uniforme, las funciones de densidad de probabilidades normal y uniforme usadas en ambos ambientes de programación pueden generar diferencias en los algoritmos que se programen como sucedió en el presente trabajo, en el caso del algoritmo Metropolis-Hastings.

2.4 Conclusiones

Se programaron en el ambiente Matlab y en lenguaje R el método frecuentista de mínimos cuadrados y el algoritmo Bayesiano Metropolis-Hastings para estimación de parámetros de un modelo dinámico de crecimiento de un cultivo de lechugas en invernadero.

Las librerías para integración numérica disponibles en ambos ambientes (*ode suite* en Matlab y *deSolve* en lenguaje R) permitieron obtener soluciones semejantes de las ecuaciones diferenciales ordinarias. La programación en ambos ambientes de la integración del modelo matemático y los métodos de estimación de parámetros, permitieron verificar el código computacional.

El modelo matemático para crecimiento de lechugas se programó correctamente como modelo computacional ya que los programas tanto en Matlab como en lenguaje R permitieron obtener los mismos resultados. Se encontró también que Matlab y R presentan tanto ventajas como desventajas de programación para resolver el problema de estimación de parámetros usando ya sea un método frecuentista o Bayesiano. Por lo tanto, es recomendable aprovechar las bondades que ambos ambientes presentan.

2.5 Referencias

Ceglar, A., Črepinšek, Z., Kajfež-Bogataj, L., and Pogačar, T. (2011). The simulation of phenological development in dynamic crop model: the Bayesian comparison of different methods. *Agricultural and Forest Meteorology* 151: 101-115.

Dzotsi, K.A., Basso, B. and Jones, J.W. (2015). Parameter and uncertainty estimation for maize, peanut and cotton using the SALUS crop model. *Agricultural Systems* 135: 31-47.

Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A. and Rubin, D.B. (2014). *Bayesian Data Analysis*. Third Edition.

Iizumi, T., Yokozawa, M., and Nishimori, M. (2009). Parameter estimation and uncertainty analysis of a large-scale crop model for paddy rice: application of a Bayesian approach. *Agricultural and Forest Meteorology* 149: 333-348.

Ioslovich, I., Seginer, I. and Baskin, A. 2002. Fitting the NICOLET lettuce growth model to plant-spacing experimental data. *Biosystems Engineering* 83(3): 361-371.

Makowski, D., Wallach, D., and Tremblay, M. (2002). Using Bayesian approach to parameter estimation; comparison of the GLUE and MCMC methods. *Agronomie* 22: 191-203.

Makowski, D., Hillier,J., Wallach, D., Andrieu, B. and Jeuffroy, M.H. 2006. Parameter estimation for crop models. En: Wallach, D.; Makowski, D. and Jones, J.W. (Editors). *Working with dynamic crop models. Evaluation, analysis, parameterization and applications*. Elsevier, Amsterdam, pp: 101-149.

Ramírez, A.A., López, C.I.L., and Rojano, A.A. (2001). Calibration of a dynamic lettuce growth model for a soilless system in a mild climate. *Proceedings of the Fourth International Symposium on Mathematical Modeling and Simulation in Agricultural and Bio-Industries* (M^2SABI'01), 12-14 June. Haifa, Israel.

van Henten. E. (1994). Validation of a dynamic lettuce growth model for greenhouse climate control. *Agricultural Systems* 45: 55-72.

van Henten E. and van Straten G. (1994). *Sensitivity analysis of a dynamic growth model of lettuce*. J. Agric. Engng. Res. 59: 19-31.

Wallach, D. 2011. Crop model calibration: a statistical perspective. *Agronomy Journal* 103(4): 1144-1151.

Wallach, D.; Makowski, D. and Jones, J.W. (Editors). 2006. *Working with dynamic crop models. Evaluation, analysis, parameterization and applications*. Elsevier, Amsterdam.

Wallach, D., Makowski, D., Jones, J.W., and Brun, F. 2014. *Working with Dynamic Crop Models. Methods, Tools and Examples for Agriculture and Environment*. Elsevier. Amsterdam. The Netherlands