

Ecuaciones en derivadas parciales como modelos en propagación de enfermedades infecciosas



María Pilar Ros Moliner
Trabajo de Fin de Grado de Matemáticas
Universidad de Zaragoza

Directores del trabajo: Francisco José Gaspar y
Carmen Rodrigo
14 de junio de 2023

Summary

Throughout history, the spread of diseases has been an issue of great importance and concern to the society, and mathematics has played a key role in understanding the factors that influence their spread. Mathematical models have allowed the estimation of key parameters such as incubation period, recovery rate, and infection rate in individuals. The aim of the dissertation is to solve mathematical models related to the spread of diseases. The models we study are ordinary differential equations system and partial differential equations systems. We focus on a parallel-in-time method that combines two different numerical methods to approximate the solution of the differential equation. It is structured as follows:

In Chapter 1, SIR model which divides the population into three groups: susceptible ($S(t)$), infected ($I(t)$) and recovered ($R(t)$) individuals, is discussed in detail. In particular, SIR model is given by the following ordinary differential equation (ODE) system.

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - \nu I(t), \\ \frac{dR(t)}{dt} = \nu I(t), \end{cases}$$

where β and ν are infection and recovery rates, respectively. In the first half of the chapter, the model is interpreted in terms of the derivative of each variable, and a comprehensive study is made of the long-term behavior of the disease. In addition, an expression that allows us to calculate the maximum number of infected individuals that a population can reach, is calculated considering the infection and recovery rates, β and ν . Following this study, the endemic SIR model, which includes births and deaths in the population, is introduced. To this end, a new variable $B(t) = \mu N(t)$ is defined, where $N(t)$ is the population size and μ is the death rate.

On the other hand, we introduce a new model that will be used throughout the rest of the chapters, that is the SIR endemic model with diffusion. This new model considers spatial mobility between individuals, so that the variables now depend on space and time. The resulting partial differential equation system is as follows

$$\begin{cases} S_t(x, t) = B(x, t) - \mu S(x, t) - \beta S(x, t)I(x, t) + D_s S_{xx}(x, t), \\ I_t(x, t) = \beta S(x, t)I(x, t) - (\nu + \mu)I(x, t) + D_I I_{xx}(x, t), \\ R_t(x, t) = \nu I(x, t) - \mu R(x, t) + D_R R_{xx}(x, t). \end{cases}$$

Finally, different variants of the model are explained depending on the immunity that individuals may develop or if the disease requires an incubation time among other factors (SI, SIS, SIRS...).

In the second chapter, the finite difference method (FDM) is introduced as a numerical tool for solving the systems of ODEs and PDEs of the target SIR models. The idea of the method is to replace the derivatives with respect to the time and space of the equations by approximations, given by the explicit Euler method in our case, that is,

$$u_{xx}(x_i, t_j) \approx \frac{1}{h^2}(u_{i-1}^j - 2u_i^j + u_{i+1}^j), \quad u_t(x_i, t_j) \approx \frac{1}{k}(u_i^{j+1} - u_i^j),$$

where u_i^j is considered as the approximation of $u(x_i, t_j)$ and h and k are the spatial and temporal discretization steps, respectively. Once the derivatives have been substituted, as we are using an explicit method, we can obtain the solution as a function of the previous time.

Besides, this chapter will also study the convergence of the method by presenting two results that guarantee stability and consistency, thus proving the convergence of this method.

Having explained the theory of the finite difference method, we can finish the chapter by explaining how to apply the finite difference method to the equations of the basic endemic SIR and endemic SIR model with diffusion. Writing the equations for a time level in terms of the previous time, we obtain different equations for each group of individuals, and that will be useful for the development of the algorithm proposed in the following chapter are presented.

In the next chapter, the parareal algorithm is presented as a parallel-in-time tool to obtain faster and more accurate solutions for ODEs and PDEs systems. The algorithm uses two different numerical methods, one high-precision temporal integrator (**F**) and one low-precision integrator (**G**). In our case, we will use the FDM with different time steps. Parareal algorithm is given as follows:

$$\begin{cases} \mathbf{u}_0^{k+1} = \mathbf{u}_0, \\ \mathbf{u}_{j+1}^{k+1} = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^{k+1}) + \mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^k) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^k) \quad j = 0, \dots, m, \end{cases}$$

where

$$\begin{cases} \mathbf{u}_0^0 = \mathbf{u}_0, \\ \mathbf{u}_{j+1}^0 = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^0) \quad j = 0, \dots, m. \end{cases}$$

In addition, three important results are presented in order to demonstrate the accuracy, convergence, and stability of the algorithm. At the end of this chapter, we take as a simple example a problem to see how the parallel algorithm is applied, and through the designed programs, we present some results to this problem.

Finally, in the last chapter of this dissertation, we present the results obtained by applying the parareal algorithm to different systems: the basic SIR model and the endemic SIR model with diffusion. For each problem, different step sizes are chosen and the parareal algorithm is run for each case. The number of iterations needed to reach convergence in each experiment is recorded and a table with the results is presented.

Índice general

Summary	III
1. Modelo SIR	1
1.1. Modelo SIR básico	2
1.1.1. Enfermedad a largo plazo	2
1.1.2. Máximo número de infectados	3
1.2. Modelo SIR endémico	4
1.2.1. Modelo SIR endémico con difusión	5
1.3. Otras variantes del modelo SIR	5
2. Método de diferencias finitas	7
2.1. Introducción al método de diferencias finitas	7
2.2. Convergencia del método	8
2.2.1. Consistencia	8
2.2.2. Estabilidad	9
2.3. Método de diferencias finitas aplicado al modelo SIR	10
3. Método parareal	15
3.1. Algoritmo parareal	15
3.2. Convergencia del algoritmo	16
3.3. Estabilidad del algoritmo	20
3.4. Simulación numérica	21
4. Aplicación del algoritmo parareal	23
4.1. Parareal para EDOs	23
4.2. Parareal para EDPs	24
Bibliografía	27
A. Códigos	29
A.1 Parareal	29
A.2 Parareal para el modelo SIR	30
A.3 Parareal para el modelo SIR endémico con difusión	31

Capítulo 1

Modelo SIR

El modelo SIR es un modelo matemático utilizado para el estudio de la propagación de enfermedades infecciosas en una población. Este modelo es propuesto en 1927 por los británicos Kermack y McKendrick con el fin del estudio de la epidemia de la peste bubónica en Bombay, India. Además, ha sido objeto de numerosas investigaciones, como por ejemplo la predicción de enfermedades como el VIH, la gripe y el SARS, al igual que se ha utilizado para la evaluación del impacto de medidas de control como el aislamiento y la vacunación.

Se considera una población en la que existe un pequeño número de personas que padecen una enfermedad infecciosa. Este modelo, que se basa en la división de la población en tres grupos: susceptibles, infectados y recuperados, estudia el número de personas que se infectarán y durante cuanto tiempo permanecerán en el grupo de individuos infectados. Existen varias extensiones del modelo básico incluyendo otros tipos de factores como el desplazamiento de la población, la incubación de la enfermedad o la inmunidad adquirida, que explicaremos más adelante.

Como ya se ha comentado anteriormente, para un tiempo t , la población total $N(t)$ se divide en 3 grupos:

- $S(t)$: Número de individuos susceptibles de contraer la enfermedad.
- $I(t)$: Número de individuos infectados, capaces de transmitir la enfermedad.
- $R(t)$: Número de individuos que ya no pueden ser infectados o han fallecido.

El modelo propuesto por Kermack y McKendrick se basa en las siguientes ecuaciones diferenciales:

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - \nu I(t), \\ \frac{dR(t)}{dt} = \nu I(t), \end{cases} \quad (1.1)$$

con las condiciones iniciales

$$\begin{aligned} S(0) &= S_0 > 0, \\ I(0) &= I_0 > 0, \\ R(0) &= 0, \end{aligned}$$

donde $\beta > 0$ y $\nu > 0$ representan los ratios de infección y recuperación, respectivamente.

El modelo se basa en las siguientes suposiciones:

1. En cuanto un individuo de la clase $S(t)$ entra en contacto con otro del grupo $I(t)$, pasa a formar parte de este último grupo. Además, los individuos de $I(t)$ dejan de formar parte de ese grupo si se recuperan, adquiriendo así la inmunidad a la enfermedad o si fallecen.

2. No existe una vacuna para la enfermedad.
3. La probabilidad de contagio en un individuo susceptible desaparece a lo largo del tiempo por el curso natural de la enfermedad, muerte o cuarentenas de los infectados, evitando el contagio a otros individuos de la población.
4. La tasa de infección β indica el número de individuos por unidad de tiempo que cambian de la clase de susceptibles a la de infectados. La tasa de recuperación ν es la tasa por la que los individuos infectados pasan a la clase de recuperados. Notar que la tasa de infección es un dato difícil de medir, mientras que la tasa de recuperación se puede calcular en un laboratorio como 1 entre la duración de la infección (D) en un individuo, es decir $\nu = \frac{1}{D}$.
5. Las tasas de recuperación y de infección son las mismas para cada individuo.
6. Para este modelo consideramos que las muertes de la población son causadas por la enfermedad ya que el número de muertes naturales es irrelevante. Debido a esto se tiene que $\frac{d(S+I+R)}{dt} = 0$, lo que supone que la población es constante, es decir, $N(t) = cte = S(t) + I(t) + R(t)$, y para el resto del trabajo asumiremos, sin pérdida de generalidad, que $N(t) = 1$.

1.1. Modelo SIR básico

Notar que la derivada $\frac{d}{dt}$ representa el cambio por unidad de tiempo de las variables de nuestro estudio. En particular, $\frac{dI(t)}{dt}$ hace referencia a la incidencia, es decir, el número de nuevos infectados por unidad de tiempo. Como $I(t) > 0$,

- $\frac{dS(t)}{dt} \leq 0$ lo que supone que el número de susceptibles decrecerá siempre que haya infectados.
- $\frac{dR(t)}{dt} \geq 0$ por lo que mientras haya infectados, aumentará el número de personas recuperadas en la población.

Notar que esto es lógico ya que toda la población forma parte del grupo de los susceptibles hasta que empiezan a aparecer infectados y por consecuencia también aumenta el número de personas recuperadas. Veamos qué ocurre para el caso de los infectados.

- Si $\frac{dI(t)}{dt} < 0 \Rightarrow \beta S(t) - \nu < 0 \Rightarrow S(t) < \frac{\nu}{\beta}$. La infección desaparecerá con el paso del tiempo.
- Si $\frac{dI(t)}{dt} > 0 \Rightarrow \beta S(t) - \nu > 0 \Rightarrow S(t) > \frac{\nu}{\beta}$. La infección aumentará con el paso del tiempo.

1.1.1. Enfermedad a largo plazo

En esta sección se estudia el modelo SIR básico, sin contemplar nacimientos ni muertes, por lo que la enfermedad termina desapareciendo con el tiempo, así se verifica que $\lim_{t \rightarrow \infty} I(t) = 0$.

Tomando

$$S_{\infty} = \lim_{t \rightarrow \infty} S(t),$$

$$R_{\infty} = \lim_{t \rightarrow \infty} R(t) = 1 - S_{\infty},$$

se llega a que el sistema (1.1) alcanza el equilibrio en el punto $(S_\infty, 0, 1 - S_\infty)$. Estudiemos ahora la estabilidad del modelo. Veamos de qué depende que el punto $(S_\infty, 0, 1 - S_\infty)$ sea estable o no, para ello, calculamos la matriz jacobiana del sistema, que resulta ser

$$J = \begin{pmatrix} -\beta I & -\beta S & 0 \\ \beta I & \beta S - \nu & 0 \\ 0 & \nu & 0 \end{pmatrix}.$$

Calculando el polinomio característico, obtenemos

$$\begin{aligned} p(\lambda) = |J - \lambda I| &= \det \begin{pmatrix} -\beta I - \lambda & -\beta S & 0 \\ \beta I & \beta S - \nu - \lambda & 0 \\ 0 & \nu & -\lambda \end{pmatrix} = -\lambda \det \begin{pmatrix} -\beta I - \lambda & -\beta S \\ \beta I & \beta S - \nu - \lambda \end{pmatrix} \\ &= -\lambda(\beta I \nu + \lambda(\beta I - \beta S + \nu + \lambda)). \end{aligned}$$

Sustituimos el punto de equilibrio en dicha expresión e igualamos a 0 para obtener los valores propios

$$p(\lambda) = -\lambda^2(-\beta S_\infty + \nu + \lambda) = 0 \Rightarrow \begin{cases} \lambda = 0 \\ \lambda = \beta S_\infty - \nu. \end{cases}$$

- Si $\lambda = \beta S_\infty - \nu < 0 \Rightarrow$ Punto estable, esto significa que a medida que el tiempo avanza, el número de personas infectadas disminuye.
- Si $\lambda = \beta S_\infty - \nu > 0 \Rightarrow$ Punto inestable, lo que supone que a lo largo del tiempo, el número de infectados aumentará hasta llegar a $\beta S_\infty - \nu < 0$.

Introducimos un ratio esencial en la epidemiología. Denotamos como r_0 al **número reproductivo básico de la enfermedad** y se expresa como $r_0 = \frac{\beta S_\infty}{\nu}$.

- Si $r_0 < 1$ la enfermedad desaparece.
- Si $r_0 > 1$ se produce un brote de la enfermedad pudiendo dar lugar a una epidemia.

1.1.2. Máximo número de infectados

El comportamiento de una enfermedad infecciosa va variando a lo largo del tiempo según las tasas de recuperación e infección, lo podemos observar en la Figura 1.1.

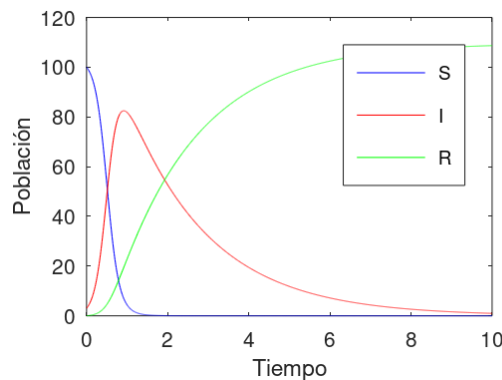


Figura 1.1: Número de individuos susceptibles (S), número de individuos infectados (I), Número de individuos recuperados (R) en un intervalo de tiempo $[0, 10]$.

Toda enfermedad alcanza un número máximo de infectados en la población y será a partir de ese momento en el que los contagios irán disminuyendo. Gracias a las condiciones iniciales y a las tasas es

posible estimar dicho máximo.

Para ello, comenzamos dividiendo las primeras ecuaciones del modelo (1.1)

$$\begin{aligned}\frac{dI}{dS} &= \frac{\beta SI - \nu I}{-\beta SI} = -1 + \frac{\nu}{\beta S}, \\ dI &= \left(-1 + \frac{\nu}{\beta S}\right) dS,\end{aligned}$$

e integrando a ambos lados de la ecuación se tiene

$$\begin{aligned}\int dI &= \int \left(-1 + \frac{\nu}{\beta S}\right) dS \iff I = -S + \frac{\nu}{\beta} \log S + C, \\ I + S - \frac{\nu}{\beta} \log S &= C.\end{aligned}$$

Luego, $\forall t \geq 0$

$$I(t) + S(t) - \frac{\nu}{\beta} \log S(t) = I(0) + S(0) - \frac{\nu}{\beta} \log S(0). \quad (1.2)$$

Notar que el máximo valor de $I(t)$ se obtiene cuando $\frac{dI}{dt} = 0$, y por la segunda ecuación del modelo, eso se cumple cuando $S(t) = \frac{\nu}{\beta}$ ya que $I(t) \neq 0$. Por lo tanto, volviendo a (1.2), $\forall t \geq 0$

$$\begin{aligned}I_{\max}(t) + \frac{\nu}{\beta} - \frac{\nu}{\beta} \log \left(\frac{\nu}{\beta}\right) &= I(0) + S(0) - \frac{\nu}{\beta} \log S(0), \\ I_{\max}(t) &= I(0) + S(0) + \frac{\nu}{\beta} \left(-\log S(0) + \log \left(\frac{\nu}{\beta}\right)\right) - \frac{\nu}{\beta}, \\ I_{\max}(t) &= I(0) + S(0) + \frac{\nu}{\beta} \log \left(\frac{\nu}{\beta S(0)}\right) - \frac{\nu}{\beta}, \\ I_{\max}(t) &= N + \frac{\nu}{\beta} \log \left(\frac{\nu}{\beta S(0)}\right) - \frac{\nu}{\beta}.\end{aligned}$$

1.2. Modelo SIR endémico

Una vez explicado el modelo más básico, es conveniente introducir una ampliación de este. El modelo SIR endémico es aquel en el que se consideran nacimientos y muertes en la población, por lo que es necesario añadir nuevos parámetros a nuestro estudio.

Se define $B(t)$ como el número de nacimientos en la población y μ como la tasa de muerte, sea natural o por la enfermedad, de esta manera el nuevo sistema es el siguiente

$$\begin{cases} \frac{dS(t)}{dt} = B(t) - \mu S(t) - \beta S(t)I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - (\nu + \mu)I(t), \\ \frac{dR(t)}{dt} = \nu I(t) - \mu R(t), \end{cases} \quad (1.3)$$

junto las condiciones iniciales

$$\begin{aligned}S(0) &= S_0 > 0, \\ I(0) &= I_0 > 0, \\ R(0) &= 0.\end{aligned}$$

1.2.1. Modelo SIR endémico con difusión

En situaciones reales, la propagación de la enfermedad se ve afectada por el movimiento espacial entre individuos y es por ello que se ha llevado a cabo una extensión del modelo SIR considerando la movilidad espacial. Para tener en cuenta la movilidad de los individuos, se introducen tantos términos nuevos como grupos de individuos tiene el modelo. En este caso, se añaden tres nuevos parámetros que representan la tasa de cambio entre los tres grupos (D_S, D_I, D_R). Asumimos que la posición espacial es unidimensional y acotada.

Sea el dominio espacial $[0, L]$, $L > 0$, las variables del modelo SIR pasan a ser dependientes del espacio y del tiempo, por lo que el nuevo sistema quedaría

$$\begin{cases} S_t(x, t) = B(x, t) - \mu S(x, t) - \beta S(x, t)I(x, t) + D_S S_{xx}(x, t), \\ I_t(x, t) = \beta S(x, t)I(x, t) - (\nu + \mu)I(x, t) + D_I I_{xx}(x, t), \\ R_t(x, t) = \nu I(x, t) - \mu R(x, t) + D_R R_{xx}(x, t), \end{cases}$$

con condiciones iniciales

$$S(x, 0) = S_0(x) > 0,$$

$$I(x, 0) = I_0(x) > 0,$$

$$R(x, 0) = 0,$$

y dado que el dominio espacial está acotado y es unidimensional, consideramos también condiciones de contorno de tipo Neumann homogéneas

$$S_x(0, t) = S_x(L, t) = I_x(0, t) = I_x(L, t) = R_x(0, t) = R_x(L, t) = 0.$$

Este modelo se estudiará con más detalle en el Capítulo 2.3.

1.3. Otras variantes del modelo SIR

Las enfermedades infecciosas se ven afectadas por distintos factores que hacen que el estudio de su propagación varíe como por ejemplo: el crecimiento demográfico, la tasa de incubación de la enfermedad o la vacunación. Es por ello que, a partir del modelo básico creado por Kermack y McKendrick, se han desarrollado nuevas variantes con el fin de abarcar todo tipo de propagación de enfermedades de manera más precisa. Presentamos los modelos más comunes.

■ Modelo SI

Se trata del modelo más simple de todos. La población queda dividida en dos subgrupos, los susceptibles (S) y los infectados (I), eliminando así el grupo de individuos recuperados ya que este modelo asume que no hay recuperación ni inmunidad.

El modelo se basa en el siguiente sistema de ecuaciones diferenciales

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t). \end{cases}$$

Se puede demostrar que este modelo garantiza que toda la población acabará infectada. En efecto, como la población total es fija, podemos escribir la segunda ecuación como

$$\frac{dI(t)}{dt} = \beta I(t)(N(t) - I(t)),$$

e integrando a ambos lados respecto t y tomando $I(0) = I_0$ se tiene

$$I(t) = \frac{I_0 N(t)}{(N(t) - I_0)e^{-\beta N(t)t} + I_0},$$

demostrando así que cuando $t \rightarrow \infty$, $I(t) \rightarrow N(t)$.

■ **Modelo SIS con nacimientos y muertes**

Este modelo contempla la posibilidad de recuperarse de la enfermedad. Sin embargo, no se obtiene la inmunidad, por tanto el individuo recuperado pasa a formar parte del grupo de individuos susceptibles.

El modelo queda descrito por las siguientes ecuaciones

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t) + B(t) - \mu S(t) + \nu I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - \mu I(t) - \nu I(t). \end{cases}$$

■ **Modelo SIRS con nacimientos y muertes**

A diferencia del modelo anterior, el modelo SIRS permite la pérdida de la inmunidad, por lo que un individuo recuperado, con el tiempo, pasa a formar parte del grupo de individuos susceptibles. Se basa en el siguiente sistema

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t) + \alpha R(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - \mu I(t) - \nu I(t), \\ \frac{dR(t)}{dt} = \nu I(t) - \mu R(t) - \alpha R(t), \end{cases}$$

donde α es la tasa promedio de pérdida de inmunidad.

■ **Modelo SEIR.**

En este modelo se introduce una nueva clase de individuos, aquellos que han estado expuestos al virus y lo están incubando $E(t)$, luego existe un nuevo ratio γ que representa la tasa de incubación de la enfermedad. Las hipótesis que se siguen son las mismas que en el modelo SIR endémico, se consideran nacimientos y muertes y los individuos una vez pasada la enfermedad adquieren inmunidad. El modelo es el siguiente:

$$\begin{cases} \frac{dS(t)}{dt} = B(t) - \mu S(t) - \beta S(t)I(t), \\ \frac{dE(t)}{dt} = \beta S(t)I(t) - (\gamma + \mu)E(t), \\ \frac{dI(t)}{dt} = \gamma E(t) - (\nu + \mu)I(t), \\ \frac{dR(t)}{dt} = \nu I(t) - \mu R(t). \end{cases}$$

Al igual que en el modelo SIR, también se puede estudiar el caso en el que existe movilidad entre los individuos. Introducimos los coeficientes de difusión para cada grupo de la población, D_S, D_E, D_I, D_R , por lo que el sistema a estudiar es el siguiente:

$$\begin{cases} S_t(x, t) = B(x, t) - \mu S(x, t) - \beta S(x, t)I(x, t) + D_S S_{xx}(x, t), \\ E_t(x, t) = \beta S(x, t)I(x, t) - (\gamma + \mu)E(x, t) + D_E E_{xx}(x, t), \\ I_t(x, t) = \gamma E(x, t) - (\nu + \mu)I(x, t) + D_I I_{xx}(x, t), \\ R_t(x, t) = \nu I(x, t) - \mu R(x, t) + D_R R_{xx}(x, t). \end{cases}$$

Cabe destacar que este modelo puede ser utilizado combinado con el resto, considerando en estos, el grupo de individuos expuestos (SEI, SEIS, SEIRS ...).

Capítulo 2

Método de diferencias finitas

2.1. Introducción al método de diferencias finitas

El método de diferencias finitas (MDF) es un método utilizado para la resolución aproximada de ecuaciones en derivadas parciales definidas en dominios finitos, en este caso espacio y tiempo. Veamos cómo plantear un esquema en diferencias finitas para el siguiente problema.

Sea $f(x, t)$ una función continua, se desea buscar una función $u(x, t)$ que verifique

$$\begin{cases} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = f(x, t) & 0 \leq t \leq T_{final}, \quad 0 \leq x \leq 1, \\ u_x(0, t) = 0, \quad u_x(1, t) = 0, \\ u(x, 0) = g(x). \end{cases} \quad (2.1)$$

Consideramos el siguiente mallado:

- Sea $n \geq 1, n \in \mathbb{N}$, se define el paso de discretización espacial como $h = \frac{1}{n+1}$, por lo que los nodos en la dirección espacial quedan definidos por $x_i = ih, i = 0, \dots, n+1$.
- Sea $m \geq 1, m \in \mathbb{N}$, se define el paso de discretización temporal como $k = \frac{T_{final}}{m+1}$, por lo que los nodos en la dirección temporal quedan definidos por $t_j = kj, j = 0, \dots, m+1$.

En el mallado anterior, consideramos u_i^j como la aproximación de $u(x_i, t_j)$.

La idea básica del método es reemplazar derivadas por fórmulas en diferencias. Por el esquema de Euler explícito, consideramos las siguientes aproximaciones

$$\begin{aligned} u_{xx}(x_i, t_j) &\approx \frac{u_{i-1}^j - 2u_i^j + u_{i+1}^j}{h^2}, \\ u_t(x_i, t_j) &\approx \frac{u_i^{j+1} - u_i^j}{k}. \end{aligned} \quad (2.2)$$

Sustituyendo estas aproximaciones en el problema (2.1), se obtiene la siguiente expresión

$$\frac{u_i^{j+1} - u_i^j}{k} - \frac{u_{i-1}^j - 2u_i^j + u_{i+1}^j}{h^2} = f(x_i, t_j). \quad (2.3)$$

Por otro lado, dado que estamos trabajando con condiciones de tipo Neumann, para aproximar la derivada respecto a x , introducimos una nueva aproximación denominada *aproximación de diferencias centrales* y se expresa como

$$u_x(x_i, t_j) \approx \frac{u_{i+1}^j - u_{i-1}^j}{2h}.$$

Es importante tener en cuenta que en los extremos del dominio, es decir, en $x = 0$ y $x = 1$, se tiene que $u_x(0, t_j) = u_x(1, t_j) = 0$, luego

$$\begin{aligned} u_x(0, t_j) &\approx \frac{u_1^j - u_{-1}^j}{2h} = 0 \Rightarrow u_{-1}^j = u_1^j, \\ u_x(1, t_j) &\approx \frac{u_{n+2}^j - u_n^j}{2h} = 0 \Rightarrow u_{n+2}^j = u_n^j. \end{aligned}$$

Estas ecuaciones muestran que los nodos u_{-1}^j y u_{n+2}^j se utilizan como nodos "fantasmas". Estos nodos ficticios se añaden al dominio para garantizar que las derivadas en los extremos se aproximen correctamente utilizando la fórmula de diferencias centrales. De esta manera, se consigue el siguiente esquema en diferencias

$$\begin{cases} \frac{u_i^{j+1} - u_i^j}{k} - \frac{u_{i-1}^j - 2u_i^j + u_{i+1}^j}{h^2} = f(x_i, t_j), & i = 0, \dots, n+1, j = 0, \dots, m+1, \\ u_{-1}^j = u_1^j, \quad u_{n+2}^j = u_n^j, \\ u_i^0 = g(x_i). \end{cases} \quad (2.4)$$

Como hemos dicho, es un método explícito ya que es posible hallar u_i^{j+1} en función de los valores del tiempo anterior. Notar que tomando $r = \frac{k}{h^2}$ se tiene

$$u_i^{j+1} = u_i^j + r(u_{i-1}^j - 2u_i^j + u_{i+1}^j) + kf(x_i, t_j), \quad i = 0, \dots, n+1, j = 0, \dots, m+1. \quad (2.5)$$

2.2. Convergencia del método

El método de diferencias finitas se utiliza para aproximar soluciones, y es por ello que es necesario estudiar la convergencia del método. Analizaremos dicha convergencia a través de la consistencia y estabilidad del método ya que son dos condiciones suficientes para verificar que la solución numérica se aproxima a la solución exacta del problema. Es decir, si se demuestra que el método es estable y consistente, se tiene que también es convergente ¹.

Teorema 2.1. Sea $u(x, t)$ la solución de (2.1), u_i^j la aproximación por el método de diferencias finitas explícito y $r = k/h^2 \leq 1/2$ entonces

$$\max_{0 \leq j \leq m+1} \max_{0 \leq i \leq n+1} |u_i^j - u(x_i, t_j)| \leq C(k + h^2).$$

Es decir,

$$\|e\|_\infty = \mathcal{O}(k + h^2),$$

siendo $e_i^j = u_i^j - u(x_i, t_j)$.

2.2.1. Consistencia

Definición. El *error de truncación* es el error que se comete al sustituir la solución numérica por la solución exacta en la ecuación de diferencias.

Definición. Un esquema de diferencias finitas se dice que es consistente de orden de precisión (p, q) si

$$\|\tau_i^j\| = \mathcal{O}(h^p) + \mathcal{O}(k^q),$$

donde τ es el error de truncación.

¹Información más detallada y los teoremas utilizados en esta sección se encuentran demostrados en [7], [8] y [9]

Teorema 2.2. *El esquema (2.4) es consistente de orden 1 en tiempo y 2 en espacio, es decir,*

$$\|\tau_i^j\| = \mathcal{O}(h^2) + \mathcal{O}(k).$$

Demostración. Comenzamos viendo cómo es el error de truncamiento

$$\tau_i^j = \frac{u(x_i, t_j + k) - u(x_i, t_j)}{k} - \frac{u(x_i - h, t_j) - 2u(x_i, t_j) + u(x_i + h, t_j)}{h^2} - f(x_i, t_j).$$

Si desarrollamos ambos términos por el desarrollo de Taylor en torno a (x_i, t_j) , obtenemos

$$\tau_i^j = \frac{\partial u(x_i, t_j)}{\partial t} + \mathcal{O}(k) - \frac{\partial^2 u(x_i, t_j)}{\partial x^2} + \mathcal{O}(h^2) - f(x_i, t_j).$$

Notar que por la ecuación del problema a estudiar (2.1), tenemos que $u_t - u_{xx} = f(x, t)$ por tanto

$$\tau_i^j = \mathcal{O}(k) + \mathcal{O}(h^2).$$

Por lo que se tiene que

$$|\tau_i^j| \leq c(k + h^2),$$

con c una constante. De esta manera queda demostrado que el método es consistente de orden uno en tiempo y orden dos en espacio. □

2.2.2. Estabilidad

Teorema 2.3. *Sea*

$$K^j = \max_{0 \leq i \leq n+1} |u_i^j|, \quad F^j = \max_{0 \leq i \leq n+1} |f(x_i, t_j)|, \quad j = 0, \dots, m+1,$$

$$\|u\|_\infty = \max_{0 \leq j \leq m+1} K^j.$$

Si $r = k/h^2 \leq 1/2$ entonces el esquema en diferencias finitas (2.4) verifica que

$$\|u\|_\infty \leq \|u^0\|_\infty + T \|F\|_\infty,$$

y por tanto, el esquema es condicionalmente estable.

Demostración. Suponemos que se han elegido los pasos de discretización temporal y espacial de tal manera que se cumpla $r = k/h^2 \leq 1/2$. Además por (2.5) se tiene

$$u_i^{j+1} = (1 - \frac{2k}{h^2})u_i^j + \frac{k}{h^2}u_{i-1}^j + \frac{k}{h^2}u_{i+1}^j + kf(x_i, t_j), \quad i = 0, \dots, n+1, \quad j = 0, \dots, m+1,$$

y dado que todos los coeficientes son positivos

$$|u_i^{j+1}| \leq (1 - \frac{2k}{h^2})|u_i^j| + \frac{k}{h^2}|u_{i-1}^j| + \frac{k}{h^2}|u_{i+1}^j| + k|f(x_i, t_j)| \leq K^j + kF^j, \quad i = 0, \dots, n+1, \quad j = 0, \dots, m+1.$$

De la ecuación anterior es fácil ver que

$$K^{j+1} \leq K^j + kF^j, \quad j = 0, \dots, m+1,$$

y

$$K^j \leq K^0 + k \sum_{l=0}^{j-1} F^l, \quad j = 0, \dots, m+1.$$

Por tanto,

$$\max_{0 \leq j \leq m+1} K^j \leq K^0 + k \sum_{l=0}^m F^l \leq K^0 + \sum_{l=0}^m \|F\|_{\infty} = K^0 + k(m+1) \|F\|_{\infty} = K^0 + T \|F\|_{\infty},$$

obteniendo,

$$\|u\|_{\infty} \leq \|u^0\|_{\infty} + T \|F\|_{\infty}.$$

Así tomando $r = k/h^2 \leq 1/2$ podemos afirmar que el método explícito es un método condicionalmente estable. \square

Finalmente, aplicando el resultado de estabilidad al error $e_i^j = u_i^j - u(x, t)$ tal que se cumple

$$\frac{e_i^{j+1} - e_i^j}{k} - \frac{e_{i-1}^j - 2e_i^j + e_{i+1}^j}{h^2} = \tau_i^j,$$

se obtiene

$$\|e\|_{\infty} \leq T \|\tau\|_{\infty},$$

demonstrando así la convergencia del método.

2.3. Método de diferencias finitas aplicado al modelo SIR

A lo largo de esta sección vamos a ver cómo aplicar el método de Euler explícito al modelo SIR básico y al modelo SIR endémico con difusión visto en el Capítulo 1.

Comenzaremos con el modelo SIR básico (1.1), el cual consiste en un sistema de ecuaciones diferenciales ordinarias (EDO). Para abordar este tipo de sistema, utilizaremos la aproximación proporcionada por el método de Euler explícito. Sea $m \in \mathbb{N}$ y $t \in [0, T_{final}]$ considerando el paso de discretización temporal $k = \frac{T_{final}}{m+1}$, utilizamos la siguiente aproximación para la derivada respecto el tiempo.

$$u'(t_j) \approx \frac{u^{j+1} - u^j}{k}, \quad \forall j = 0, \dots, m+1.$$

Luego sea el sistema del modelo SIR básico,

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - \nu I(t), \\ \frac{dR(t)}{dt} = \nu I(t), \\ S(0) = S_0 > 0, \quad I(0) = I_0 > 0, \quad R(0) = 0. \end{cases}$$

podemos expresar las tres ecuaciones de la siguiente manera:

Susceptibles

$$\begin{aligned} \frac{S^{j+1} - S^j}{k} &= -\beta S^j I^j, \\ S^{j+1} &= S^j (1 - \beta k S^j I^j). \end{aligned}$$

Infectados

$$\begin{aligned}\frac{I^{j+1} - I^j}{k} &= \beta S^j I^j - \nu I^j, \\ I^{j+1} &= I^j(1 + \beta k S^j - k\nu).\end{aligned}$$

Recuperados

$$\begin{aligned}\frac{R^{j+1} - R^j}{k} &= \nu I^j, \\ R^{j+1} &= R^j + k\nu I^j.\end{aligned}$$

Por lo tanto, el sistema (1.1) del modelo SIR, tras aplicarle el método de Euler explícito quedaría de la siguiente manera

$$\begin{cases} S^{j+1} = S^j(1 - \beta k S^j I^j), \\ I^{j+1} = I^j(1 + \beta k S^j - k\nu), \\ R^{j+1} = R^j + k\nu I^j, \\ S^0 = S_0, \quad I^0 = I_0, \quad R^0 = 0. \end{cases} \quad (2.6)$$

Por último, veamos cómo sería el esquema en diferencias al aplicar el método de Euler explícito al modelo SIR endémico con difusión.

Sea $(x, t) \in [0, 1] \times [0, T_{final}]$, el problema a estudiar es el siguiente

$$\begin{cases} S_t(x, t) = B(x, t) - \mu S - \beta S(x, t)I(x, t) + D_s S_{xx}(x, t), \\ I_t(x, t) = \beta S(x, t)I(x, t) - (\nu + \mu)I(x, t) + D_I I_{xx}(x, t), \\ R_t(x, t) = \nu I(x, t) - \mu R(x, t) + D_R R_{xx}(x, t), \\ S(x, 0) = S_0(x), \quad I(x, 0) = I_0(x), \quad R(x, 0) = 0, \\ S_x(0, t) = S_x(1, t) = 0, \\ I_x(0, t) = I_x(1, t) = 0, \\ R_x(0, t) = R_x(1, t) = 0. \end{cases} \quad (2.7)$$

Tomamos el mallado $\{(x_i, t_j) = (ih, jk), i = 0, 1, \dots, n+1, j = 0, 1, \dots, m+1\}$, con paso de discretización espacial $h = \frac{1}{n+1}$ y temporal $k = \frac{T_{final}}{m+1}$. El objetivo es aplicar el MDF a cada clase de individuos por separado.

Susceptibles

Empezaremos con la primera ecuación de (2.7), la cual hace referencia al grupo de individuos susceptibles en la población. Luego el problema a estudiar es el siguiente

$$\begin{cases} S_t(x, t) = D_s S_{xx}(x, t) + B(x, t) - \mu S(x, t) - \beta S(x, t)I(x, t), \\ S(x, 0) = S_0, \\ S_x(0, t) = S_x(1, t) = 0. \end{cases} \quad (2.8)$$

Recordamos que por el método de Euler explícito, consideramos las siguientes aproximaciones

$$\begin{aligned}S_t(x_i, t_j) &\approx \frac{S_i^{j+1} - S_i^j}{k}, \\ S_{xx}(x_i, t_j) &\approx \frac{S_{i-1}^j - 2S_i^j + S_{i+1}^j}{h^2}, \\ S_x(x_i, t_j) &\approx \frac{S_{i+1}^j - S_{i-1}^j}{2h}.\end{aligned}$$

Por lo tanto, podemos sustituir dichas aproximaciones en la ecuación del problema (2.8). Luego, para $i = 0, \dots, n+1$ y $j = 0, \dots, m+1$ se tiene

$$\begin{aligned}\frac{S_i^{j+1} - S_i^j}{k} &= D_S \frac{S_{i-1}^j - 2S_i^j + S_{i+1}^j}{h^2} + B_i^j - \mu S_i^j - \beta S_i^j I_i^j, \\ S_i^{j+1} - S_i^j &= \frac{kD_S}{h^2} (S_{i-1}^j - 2S_i^j + S_{i+1}^j) + kB_i^j - k\mu S_i^j - k\beta S_i^j I_i^j, \\ S_i^{j+1} &= (1 - \mu k - 2D_S \frac{k}{h^2}) S_i^j + D_S \frac{k}{h^2} S_{i-1}^j + D_S \frac{k}{h^2} S_{i+1}^j + kB_i^j - k\beta S_i^j I_i^j.\end{aligned}$$

Además por las condiciones de tipo Neumann,

$$\begin{aligned}S_x(0, t_j) &\approx \frac{S_1^j - S_{-1}^j}{2h} = 0 \Rightarrow S_{-1}^j = S_1^j, \\ S_x(1, t_j) &\approx \frac{S_{n+2}^j - S_n^j}{2h} = 0 \Rightarrow S_{n+2}^j = S_n^j.\end{aligned}$$

Por lo que para la clase de individuos susceptibles (S), tomando $r = \frac{k}{h^2}$ debemos resolver

$$\begin{cases} S_i^{j+1} = (1 - \mu k - 2D_S r) S_i^j + D_S r S_{i-1}^j + D_S r S_{i+1}^j + kB_i^j - k\beta S_i^j I_i^j, & i = 0, \dots, n+1, \quad j = 0, \dots, m+1, \\ S_{-1}^j = S_1^j, \\ S_{n+2}^j = S_n^j. \end{cases} \quad (2.9)$$

Infectados

De la misma manera, utilizando el método explícito, hallamos la ecuación para el grupo de individuos infectados.

Sea

$$\begin{cases} I_t(x, t) = D_I I_{xx}(x, t) - (\nu + \mu) I(x, t) + \beta S(x, t) I(x, t), \\ I(x, 0) = I_0, \quad R(x, 0) = 0, \\ I_x(0, t) = I_x(1, t) = 0, \end{cases}$$

y teniendo en cuenta que las aproximaciones son las siguientes

$$\begin{aligned}I_t(x_i, t_j) &\approx \frac{I_i^{j+1} - I_i^j}{k}, \\ I_{xx}(x_i, t_j) &\approx \frac{I_{i-1}^j - 2I_i^j + I_{i+1}^j}{h^2}, \\ I_x(x_i, t_j) &\approx \frac{I_{i+1}^j - I_{i-1}^j}{2h},\end{aligned}$$

se tiene que para $i = 0, \dots, n+1$ y $j = 0, \dots, m+1$

$$\begin{aligned}\frac{I_i^{j+1} - I_i^j}{k} &= D_I \frac{I_{i-1}^j - 2I_i^j + I_{i+1}^j}{h^2} - (\nu + \mu) I_i^j + \beta S_i^j I_i^j, \\ I_i^{j+1} - I_i^j &= D_I \frac{k}{h^2} (I_{i-1}^j - 2I_i^j + I_{i+1}^j) - k(\nu + \mu) I_i^j + k\beta S_i^j I_i^j, \\ I_i^{j+1} &= (1 - 2D_I \frac{k}{h^2} - k(\nu + \mu)) I_i^j + D_I \frac{k}{h^2} I_{i-1}^j + D_I \frac{k}{h^2} I_{i+1}^j + k\beta S_i^j I_i^j.\end{aligned}$$

Por otro lado, por las condiciones de tipo Neumann,

$$\begin{aligned}I_x(0, t_j) &\approx \frac{I_1^j - I_{-1}^j}{2h} = 0 \Rightarrow I_{-1}^j = I_1^j, \\ I_x(1, t_j) &\approx \frac{I_{n+2}^j - I_n^j}{2h} = 0 \Rightarrow I_{n+2}^j = I_n^j.\end{aligned}$$

Así, tomando $r = \frac{k}{h^2}$ obtenemos

$$\begin{cases} I_i^{j+1} = (1 - 2D_I r - k(v + \mu))I_i^j + D_I r I_{i-1}^j + D_I r I_{i+1}^j + k\beta S_i^j I_i^j, & i = 0, \dots, n+1, \quad j = 0, \dots, m+1, \\ I_{-1}^j = I_1^j, \\ I_{n+2}^j = I_n^j. \end{cases} \quad (2.10)$$

Recuperados

Siguiendo el mismo proceso, veamos cómo es la ecuación para los individuos recuperados. Recordamos

$$\begin{cases} R_t(x, t) = D_R R_{xx}(x, t) + vI(x, t) - \mu R(x, t), \\ R(x, 0) = 0, \\ R_x(0, t) = R_x(1, t) = 0. \end{cases}$$

Por el método de Euler explícito,

$$\begin{aligned} R_t(x_i, t_j) &\approx \frac{R_i^{j+1} - R_i^j}{k}, \\ R_{xx}(x_i, t_j) &\approx \frac{R_{i-1}^j - 2R_i^j + R_{i+1}^j}{h^2}, \\ R_x(x_i, t_j) &\approx \frac{R_{i+1}^j - R_{i-1}^j}{2h}, \end{aligned}$$

luego para $i = 0, \dots, n+1$ y $j = 0, \dots, m+1$, se tiene

$$\begin{aligned} \frac{R_i^{j+1} - R_i^j}{k} &= D_R \frac{R_{i-1}^j - 2R_i^j + R_{i+1}^j}{h^2} + vI_i^j - \mu R_i^j, \\ R_i^{j+1} - R_i^j &= D_R \frac{k}{h^2} (R_{i-1}^j - 2R_i^j + R_{i+1}^j) + kvI_i^j - k\mu R_i^j, \\ R_i^{j+1} &= (1 - 2D_R \frac{k}{h^2} - k\mu)R_i^j + D_R \frac{k}{h^2} R_{i-1}^j + D_R \frac{k}{h^2} R_{i+1}^j + kvI_i^j. \end{aligned}$$

Además por las condiciones de tipo Neumann

$$\begin{aligned} R_x(0, t_j) &\approx \frac{R_1^j - R_{-1}^j}{2h} = 0 \Rightarrow R_{-1}^j = R_1^j, \\ R_x(1, t_j) &\approx \frac{R_{n+2}^j - R_n^j}{2h} = 0 \Rightarrow R_{n+2}^j = R_n^j. \end{aligned}$$

Por lo que tomando $r = \frac{k}{h^2}$, para el grupo de individuos recuperados se tiene

$$\begin{cases} R_i^{j+1} = (1 - 2D_R r - k\mu)R_i^j + D_R r R_{i-1}^j + D_R r R_{i+1}^j + kvI_i^j, & i = 0, \dots, n+1, \quad j = 0, \dots, m+1, \\ R_{-1}^j = R_1^j, \\ R_{n+2}^j = R_n^j. \end{cases} \quad (2.11)$$

Por lo tanto, por (2.9), (2.10) y (2.11), el esquema en diferencias del modelo SIR endémico con difusión lo podemos expresar de la siguiente manera.

Para $i = 0, \dots, n+1$ y $j = 0, \dots, m+1$,

$$\begin{cases} S_i^{j+1} = (1 - \mu k - 2D_S r)S_i^j + D_S r S_{i-1}^j + D_S r S_{i+1}^j + kB - k\beta S_i^j I_i^j, \\ I_i^{j+1} = (1 - 2D_I r - k(v + \mu))I_i^j + D_I r I_{i-1}^j + D_I r I_{i+1}^j + k\beta S_i^j I_i^j, \\ R_i^{j+1} = (1 - 2D_R r - k\mu)R_i^j + D_R r R_{i-1}^j + D_R r R_{i+1}^j + kvI_i^j, \\ S_{-1}^j = S_1^j, \quad I_{-1}^j = I_1^j, \quad R_{-1}^j = R_1^j, \\ S_{n+2}^j = S_n^j, \quad I_{n+2}^j = I_n^j, \quad R_{n+2}^j = R_n^j. \end{cases} \quad (2.12)$$

Capítulo 3

Método parareal

En 2001, Roger E. Lions, Yvon Maday y Gabriel Turinici propusieron por primera vez el algoritmo parareal en su artículo titulado “*Résolution d’EDP par un schéma en temps pararéel*”. El algoritmo parareal es una técnica numérica paralela en tiempo para obtener soluciones más rápidas y más precisas para un sistema de ecuaciones diferenciales ordinarias (EDOs) o ecuaciones diferenciales parciales (EDPs) de forma iterativa. Utiliza dos métodos numéricos diferentes para aproximar la solución del sistema, uno de alta precisión (**F**) y otro de baja precisión (**G**). La solución se calcula mediante una combinación de ambas soluciones.

A lo largo de este capítulo comenzaremos explicando el algoritmo, veremos posteriormente dos teoremas de suma importancia sobre su convergencia y error y por último, estudiaremos su estabilidad. Las demostraciones sobre este algoritmo las podemos encontrar más detalladas en [10] y [11].

3.1. Algoritmo parareal

Sea el problema

$$\begin{cases} u'(t) = f(t, u(t)), & t \in (0, T] \\ u(0) = u_0. \end{cases} \quad (3.1)$$

Consideramos una partición del espacio temporal en $m + 1$ subintervalos $(t_j, t_{j+1}]$ con $j = 0, \dots, m$, tomando el inicio del tiempo como t_0 y T como el final. Por lo que los subintervalos se definen de manera que $0 = t_0 < t_1 < \dots < t_m < t_{m+1} = T$. Así el problema (3.1) queda dividido en varios subproblemas, por lo que el objetivo será hallar la solución de cada problema para cada intervalo de tiempo. En particular, el algoritmo parareal combina la aproximación de la solución de cada subproblema de (3.1) obtenida mediante un método poco preciso y la solución del mismo obteniéndola a través de otro método de alta precisión.

Sea la condición inicial de cada subproblema $u(t_j) = u_j$, para $j = 0, \dots, m + 1$ definimos

- $\mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j)$ como el operador con el que se obtiene una aproximación poco precisa de $u(t_{j+1})$ pero que tiene un bajo costo computacional.
- $\mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j)$ será aquel con el que conseguimos una solución mucho más precisa de $u(t_{j+1})$ pero con un mayor costo computacional.

La elección del método para las aproximaciones de la solución dependerá de varios factores, como por ejemplo el tipo de problema que se esté resolviendo o la precisión de solución requerida. En nuestro caso, estamos trabajando con un sistema de EDOs, por lo que, podemos utilizar el método de Euler tanto implícito como explícito o un método Runge-Kutta.

El algoritmo parte de los siguientes valores obtenidos a través de **G** en todo el intervalo $(0, T]$,

$$\mathbf{u}_0^0 = u_0, \quad \mathbf{u}_{j+1}^0 = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^0) \quad \forall j = 0, \dots, m. \quad (3.2)$$

De esta manera, se va obteniendo un conjunto de soluciones $\{\mathbf{u}_j^0\}_{j=0,\dots,m+1}$ el cual se debe corregir ya que, al tratarse de un método poco preciso, no es una buena aproximación de la solución del problema (3.1). Es por ello que se introduce el operador \mathbf{F} nombrado anteriormente y se define

$$\tilde{\mathbf{u}}_{j+1}^0 = \mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^0) \quad \forall j = 0, \dots, m.$$

Así, podemos escribir la corrección de la solución como

$$\Delta_{j+1}^0 = \tilde{\mathbf{u}}_{j+1}^0 - \mathbf{u}_{j+1}^0.$$

Una vez hallada la diferencia Δ_{j+1}^0 , podemos calcular la siguiente iteración del algoritmo, que vendrá dada por $\mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^1)$ y la corrección Δ_{j+1}^0 . Es decir, $\forall j = 0, \dots, m$ se tiene:

$$\mathbf{u}_{j+1}^1 = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^1) + \Delta_{j+1}^0 = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^1) + \mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^0) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^0),$$

y teniendo en cuenta que en el tiempo t_0 la solución vendrá dada por el problema (3.1), tenemos $\mathbf{u}_0^1 = \mathbf{u}_0$. Por lo que, repitiendo el proceso anterior, en términos generales, para cada iteración del algoritmo se sigue que

$$\begin{cases} \mathbf{u}_0^{k+1} = \mathbf{u}_0, \\ \mathbf{u}_{j+1}^{k+1} = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^{k+1}) + \mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^k) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^k) \end{cases} \quad j = 0, \dots, m. \quad (3.3)$$

donde

$$\begin{cases} \mathbf{u}_0^0 = \mathbf{u}_0, \\ \mathbf{u}_{j+1}^0 = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j) \end{cases} \quad j = 0, \dots, m. \quad (3.4)$$

Es importante notar que los términos $\mathbf{u}_0^0, \dots, \mathbf{u}_{m+1}^0$ son calculados de manera secuencial partiendo de la solución inicial $\mathbf{u}_0 = \mathbf{u}_0^0$, mientras que $\tilde{\mathbf{u}}$ se halla a través del integrador \mathbf{F} de forma paralela ya que \mathbf{F} no requiere información de la iteración actual.

3.2. Convergencia del algoritmo

A continuación introducimos dos resultados de gran importancia. Lo primero es importante garantizar que el método converge si el procesador \mathbf{F} es lo suficiente preciso y para ello se presenta el siguiente teorema.

Teorema 3.1. *El algoritmo parareal definido por (3.3) y (3.4) cumple la siguiente propiedad*

$$\mathbf{u}_j^k = \mathbf{F}(0, t_j, \mathbf{u}_0), \quad \forall k \geq j, \quad (3.5)$$

es decir, el vector \mathbf{u}_j^k coincide con el propagador \mathbf{F} para todas las iteraciones empezando desde $k = j$.

Demostración. Dicho resultado se demuestra por inducción sobre j . Sea $j = 0$, por (3.2) sabemos que $\mathbf{u}_0^0 = \mathbf{u}_0$ y por (3.3) se tiene que $\forall k \geq 1$, $\mathbf{u}_0^k = \mathbf{u}_0$. Además $t_0 = 0$, por lo que se cumple $\mathbf{F}(0, t_0, \mathbf{u}_0) = \mathbf{F}(0, 0, \mathbf{u}_0)$, y la igualdad (3.5) es cierta $\forall k \geq j = 0$.

Supongamos por inducción que $\mathbf{u}_j^k = \mathbf{F}(0, t_j, \mathbf{u}_0)$ se cumple $\forall k \geq j$.

Veamos si $\mathbf{u}_j^k = \mathbf{F}(0, t_j, \mathbf{u}_0)$ se cumple para $k \geq j + 1$. Por (3.3), se tiene

$$\mathbf{u}_{j+1}^k = \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^k) + \mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^{k-1}) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^{k-1}), \quad j = 0, \dots, m, \quad (3.6)$$

y además notar que $k - 1 \geq j$, luego por hipótesis de inducción se cumple

$$\mathbf{u}_j^k = \mathbf{u}_j^{k-1} = \mathbf{F}(0, t_j, \mathbf{u}_0).$$

Sustituyendo esta última expresión en (3.6), obtenemos

$$\begin{aligned}\mathbf{u}_{j+1}^k &= \mathbf{G}(t_j, t_{j+1}, \mathbf{F}(0, t_j, \mathbf{u}_0)) + \mathbf{F}(t_j, t_{j+1}, \mathbf{F}(0, t_j, \mathbf{u}_0)) - \mathbf{G}(t_j, t_{j+1}, \mathbf{F}(0, t_j, \mathbf{u}_0)) = \\ &= \mathbf{F}(t_j, t_{j+1}, \mathbf{F}(0, t_j, \mathbf{u}_0)).\end{aligned}$$

Notar que dado que $\mathbf{F}(t_j, t_{j+1}, \mathbf{F}(0, t_j, \mathbf{u}_0))$ se ha calculado empezando desde el vector $\mathbf{F}(0, t_j, \mathbf{u}_0)$ a través de \mathbf{F} , al final, el proceso completo es lo mismo que usar el propagador \mathbf{F} desde 0 a t_{j+1} con el vector inicial \mathbf{u}_0 , luego se tiene

$$\mathbf{u}_{j+1}^k = \mathbf{F}(t_j, t_{j+1}, \mathbf{F}(0, t_j, \mathbf{u}_0)) = \mathbf{F}(0, t_{j+1}, \mathbf{u}_0),$$

demostrando así que la igualdad (3.5) se cumple $\forall k \geq j+1$. \square

Otro teorema relevante es el que vamos a presentar a continuación, el cual garantiza la convergencia bajo ciertas hipótesis al igual que nos proporciona una cota para el error.

Antes de introducir el teorema, consideramos la longitud de los subintervalos del dominio temporal uniforme como $\sigma = t_{j+1} - t_j$ para todo $j = 0, \dots, m$.

Teorema 3.2. (*Estimación de Convergencia del método parareal.*) Sea \mathbf{F} la solución exacta de (3.1) y sea \mathbf{G} la solución aproximada con error de truncamiento acotado por $C_3\sigma^{p+1}$ y que satisface la condición de Lipschitz

$$\|\mathbf{G}(t, t + \sigma, \mathbf{v}) - \mathbf{G}(t, t + \sigma, \mathbf{w})\| \leq (1 + C_2\sigma) \|\mathbf{v} - \mathbf{w}\|. \quad (3.7)$$

Además, si la diferencia entre la aproximación y la solución exacta puede desarrollarse para un σ suficientemente pequeño, es decir, si

$$\mathbf{F}(t_{j-1}, t_j, \mathbf{x}) - \mathbf{G}(t_{j-1}, t_j, \mathbf{x}) = c_{p+1}(\mathbf{x})\sigma^{p+1} + c_{p+2}(\mathbf{x})\sigma^{p+2} + \dots \quad (3.8)$$

siendo los c_n funciones de clase \mathcal{C}^1 para $n = p+1, p+2, \dots$, entonces para la iteración k del algoritmo, se tiene la siguiente cota

$$\begin{aligned}\|\mathbf{u}(t_j) - \mathbf{u}_j^k\| &\leq \frac{C_3}{C_1} \frac{(C_1\sigma^{p+1})^{k+1}}{(k+1)!} (1 + C_2\sigma)^{j-k-1} \prod_{l=0}^k (j-l) \\ &\leq \frac{C_3}{C_1} \frac{(C_1 t_j)^{k+1}}{(k+1)!} e^{C_2(t_j - t_{k+1})} \sigma^{p(k+1)},\end{aligned} \quad (3.9)$$

donde C_1 es una constante dependiente de los c_n .

Demostración. Una de las hipótesis del problema es que \mathbf{F} es solución exacta del problema (3.1), por tanto se cumple $\mathbf{u}(t_{j+1}) = \mathbf{F}(t_j, t_{j+1}, \mathbf{u}(t_j))$. Luego teniendo en cuenta (3.3), se tiene

$$\mathbf{u}(t_{j+1}) - \mathbf{u}_{j+1}^{k+1} = \mathbf{F}(t_j, t_{j+1}, \mathbf{u}(t_j)) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_{j+1}^{k+1}) - \mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^k) + \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^k), \quad j = 0, \dots, m.$$

Sumando y restando $\mathbf{G}(t_j, t_{j+1}, \mathbf{u}(t_j))$ y reordenando términos obtenemos la siguiente expresión

$$\begin{aligned}\mathbf{u}(t_{j+1}) - \mathbf{u}_{j+1}^{k+1} &= \mathbf{F}(t_j, t_{j+1}, \mathbf{u}(t_j)) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}(t_j)) \\ &\quad - (\mathbf{F}(t_j, t_{j+1}, \mathbf{u}_j^k) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^k)) \\ &\quad + (\mathbf{G}(t_j, t_{j+1}, \mathbf{u}(t_j)) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^{k+1})).\end{aligned}$$

Analizando cada término por separado y haciendo uso de la expresión (3.8), se tiene

$$\begin{aligned}\mathbf{u}(t_{j+1}) - \mathbf{u}_{j+1}^{k+1} &= c_{p+1}(\mathbf{u}(t_j))\sigma^{p+1} + c_{p+2}(\mathbf{u}(t_j))\sigma^{p+2} + \dots \\ &\quad - (c_{p+1}(\mathbf{u}_j^k)\sigma^{p+1} + c_{p+2}(\mathbf{u}_j^k)\sigma^{p+2} + \dots) \\ &\quad + (\mathbf{G}(t_j, t_{j+1}, \mathbf{u}(t_j)) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^{k+1})).\end{aligned} \quad (3.10)$$

Además notar que los c_n son de clase \mathcal{C}^1 , por lo que existe una constante C_1 tal que se tiene

$$\|(c_{p+1}(\mathbf{u}(t_j)) - c_{p+1}(\mathbf{u}_j^k))\sigma^{p+1} + (c_{p+2}(\mathbf{u}(t_j)) - c_{p+2}(\mathbf{u}_j^k))\sigma^{p+2} + \dots\| \leq C_1 \sigma^{p+1} \|\mathbf{u}(t_j) - \mathbf{u}_j^k\|.$$

Así, tomando normas en (3.10) y utilizando esta última expresión y la condición de Lipschitz (3.7) se tiene

$$e_{j+1}^{k+1} := \|\mathbf{u}(t_{j+1}) - \mathbf{u}_{j+1}^{k+1}\| \leq C_1 \sigma^{p+1} \|\mathbf{u}(t_j) - \mathbf{u}_j^k\| + (1 + C_2 \sigma) \|\mathbf{u}(t_j) - \mathbf{u}_j^{k+1}\|,$$

donde se ha denotado así el error con el objetivo de definir la siguiente relación de recurrencia

$$e_{j+1}^{k+1} = \alpha e_j^k + \beta e_j^{k+1}, \quad (3.11)$$

donde $\alpha := C_1 \sigma^{p+1}$ y $\beta := (1 + C_2 \sigma)$. Para la inicialización de (3.11), tomamos (3.4) del algoritmo parareal y seguimos con el mismo procedimiento que antes

$$\begin{aligned} \|\mathbf{u}(t_{j+1}) - \mathbf{u}_{j+1}^0\| &= \|\mathbf{u}(t_{j+1}) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^0)\| = \|\mathbf{u}(t_{j+1}) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}(t_j)) + \mathbf{G}(t_j, t_{j+1}, \mathbf{u}(t_j)) - \mathbf{G}(t_j, t_{j+1}, \mathbf{u}_j^0)\| \\ &\leq C_3 \sigma^{p+1} + (1 + C_2 \sigma) \|\mathbf{u}(t_j) - \mathbf{u}_j^0\|. \end{aligned}$$

Luego tomando $\gamma := C_3 \sigma^{p+1}$, podemos definir la inicialización de la relación de recurrencia (3.11) como

$$e_{j+1}^0 = \gamma + \beta e_j^0. \quad (3.12)$$

Para analizar dicha relación, multiplicamos (3.11) por ζ^{j+1} , sumamos sobre j , y como se tiene que $e_0^k := \|\mathbf{u}(0) - \mathbf{u}_0^k\|$ podemos comenzar los sumatorios desde $j = 1$, por lo que se tiene

$$\begin{aligned} \sum_{j=1}^{\infty} e_{j+1}^{k+1} \zeta^{j+1} &= \alpha \sum_{j=1}^{\infty} e_j^k \zeta^{j+1} + \beta \sum_{j=1}^{\infty} e_j^{k+1} \zeta^{j+1} \\ &= \alpha \zeta \sum_{j=1}^{\infty} e_j^k \zeta^j + \beta \zeta \sum_{j=1}^{\infty} e_j^{k+1} \zeta^j. \end{aligned}$$

Luego tomando la función generadora $\rho_k(\zeta) := \sum_{j=1}^{\infty} e_j^k \zeta^j$, se tiene que se cumple

$$\rho_{k+1}(\zeta) = \alpha \zeta \rho_k(\zeta) + \beta \zeta \rho_{k+1}(\zeta),$$

y por tanto se tiene

$$\rho_{k+1}(\zeta) = \frac{\alpha^{k+1} \zeta^{k+1}}{(1 - \beta \zeta)^{k+1}} \rho_0(\zeta). \quad (3.13)$$

Para hallar $\rho_0(\zeta)$, tomamos la inicialización (3.12) y seguimos el mismo proceso que antes, por lo que se tiene

$$\sum_{j=0}^{\infty} e_{j+1}^0 \zeta^{j+1} = \gamma \zeta \sum_{j=0}^{\infty} \zeta^j + \beta \zeta \sum_{j=0}^{\infty} e_j^0 \zeta^j,$$

luego

$$\begin{aligned} \rho_0(\zeta) &= \gamma \frac{\zeta}{1 - \zeta} + \beta \zeta \rho_0(\zeta), \\ \rho_0(\zeta) &= \frac{\gamma \zeta}{(1 - \zeta)(1 - \beta \zeta)}. \end{aligned}$$

Sustituyendo esta expresión en (3.13),

$$\rho_k(\zeta) = \gamma \alpha^k \frac{\zeta^{k+1}}{(1 - \zeta)(1 - \beta \zeta)^{k+1}}.$$

Notar que los coeficientes de esta expresión serán los que acoten el error del algoritmo parareal, así que veamos cómo calcularlos.

Es conveniente escribir $\frac{1}{1-\zeta}$ y $\frac{1}{1-\beta\zeta}$ como series de potencias, esto es

$$\begin{aligned}\frac{1}{1-\zeta} &= 1 + \zeta + \zeta^2 + \dots \\ \frac{1}{1-\beta\zeta} &= 1 + \beta\zeta + \beta^2\zeta^2 + \dots\end{aligned}$$

Además notar que $\beta \geq 1$, luego podemos modificar la función generatriz $\rho_k(\zeta)$ como

$$\tilde{\rho}_k(\zeta) = \gamma\alpha^k \frac{\zeta^{k+1}}{(1-\beta\zeta)^{k+2}}.$$

Además, por la fórmula de la serie binomial podemos escribir

$$\frac{1}{(1-\beta\zeta)^{k+2}} = \sum_{n=0}^{\infty} \binom{k+1+n}{n} \beta^n \zeta^n.$$

Luego se tiene que

$$\tilde{\rho}_k(\zeta) = \gamma\alpha^k \sum_{n=0}^{\infty} \binom{k+1+n}{n} \beta^n \zeta^{k+1+n} = \gamma\alpha^k \sum_{j=k+1}^{\infty} \binom{j}{j-k-1} \beta^{j-k-1} \zeta^j.$$

Notar que para $j \leq k$, los coeficientes son nulos. Además, podemos escribir $\tilde{\rho}_k(\zeta)$ de la siguiente manera

$$\tilde{\rho}_k(\zeta) = \frac{\gamma\alpha^k}{(k+1)!} \sum_{j=k+1}^{\infty} \prod_{l=0}^k (j-l) \beta^{j-k-1} \zeta^j.$$

Observar que para $j > k$

$$e_j^k \leq \frac{\gamma\alpha^k}{(k+1)!} \beta^{j-k-1} \prod_{l=0}^k (j-l),$$

y además también se ve que esta desigualdad se cumple para $j \leq k$ ya que el productorio es nulo para $k > j$.

Finalmente, teniendo en cuenta que el error del algoritmo está acotado por e_j^k , se sigue

$$\begin{aligned}\|\mathbf{u}(t_j) - \mathbf{u}_j^k\| &\leq \frac{\gamma\alpha^k}{(k+1)!} \beta^{j-k-1} \prod_{l=0}^k (j-l) \\ &= \frac{C_3}{C_1} \frac{(C_1\sigma^{p+1})^{k+1}}{(k+1)!} (1 + C_2\sigma)^{j-k-1} \prod_{l=0}^k (j-l),\end{aligned}\tag{3.14}$$

demostrando así la primera parte del teorema.

Para la segunda parte, se debe tener en cuenta que $1 + C_2\sigma \leq e^{C_2\sigma}$ y además

$$(C_1\sigma^{p+1})^{k+1} \prod_{l=0}^k (j-l) \leq \sigma^{p(k+1)} (C_1\sigma)^{k+1} j^{k+1} = \sigma^{p(k+1)} (C_1 t_j)^{k+1}.$$

Por tanto, por (3.14), obtenemos el resultado

$$\|\mathbf{u}(t_j) - \mathbf{u}_j^k\| \leq \frac{C_3}{C_1} \frac{(C_1 t_j)^{k+1}}{(k+1)!} e^{C_2(t_j - t_{k+1})} \sigma^{p(k+1)}.$$

□

3.3. Estabilidad del algoritmo

Finalmente, en esta sección se demuestra un teorema que garantiza la estabilidad del algoritmo. Encontramos información mas detallada en [10].

Sea el problema

$$\begin{cases} u'(t) = \lambda f(t, u(t)), & \lambda \in \mathbb{R}, \quad t \in (0, T] \\ u(0) = u_0, \end{cases} \quad (3.15)$$

consideramos que la longitud σ es dividida en subintervalos de longitud δt , luego σ queda dividido en $s := \frac{\sigma}{\delta t}$ intervalos. Para estudiar la estabilidad del algoritmo debemos obtener la función de estabilidad $H(j, k, r, R)$ tal que cumple $u_j^k = H(j, k, r, R)u_0$, con j el índice temporal, k índice de iteración, r la función de estabilidad para el integrador \mathbf{F} y R la función de estabilidad para el integrador \mathbf{G} . Por tanto, será necesario resolver una doble relación de recurrencia como se ha hecho a lo largo de la sección anterior. Tomando la inicialización como $u_j^0 = R(\lambda \sigma)$ y sea $\bar{r} = r(\lambda \delta t)^s$ se llega a que podemos definir

$$H(j, k, r, R) = \sum_{i=0}^k \binom{j}{i} (\bar{r} - R)^i R^{j-i}. \quad (3.16)$$

Sabemos que para que la estabilidad se alcance se debe cumplir

$$\sup_{1 \leq j \leq m+1} \sup_{1 \leq k \leq m+1} |H(j, k, r, R)| \leq 1.$$

Veamos bajo qué condiciones se cumple la desigualdad anterior.

Tomando valores absolutos

$$|H(j, k, r, R)| \leq \sum_{i=0}^k \binom{n}{i} |(\bar{r} - R)|^i |R|^{j-i} \leq \sum_{i=0}^j \binom{n}{i} |(\bar{r} - R)|^i |R|^{j-i} \leq (|\bar{r} - R| + |R|)^j,$$

por tanto si se cumple $(|\bar{r} - R| + |R|)^j \leq 1$, se tendrá que $|H(j, k, r, R)| \leq 1$.

Además, notar que $|\bar{r} - R| + |R| = |\bar{r}|$ ó $|\bar{r} - R| + |R| = |\bar{r} - 2R|$, luego

■ $|\bar{r}| \leq 1$ se cumplirá si la condición de estabilidad se satisface para el operador \mathbf{F} .

■ $|\bar{r} - 2R| \leq 1$ se cumplirá si se satisface la ecuación $\frac{\bar{r} - 1}{2} \leq R \leq \frac{\bar{r} + 1}{2}$.

Una vez explicado esto, podemos introducir el siguiente teorema sobre la estabilidad.

Teorema 3.3. Sean $r = r(\lambda \delta t)$ y $R = R(\lambda \sigma)$ las funciones de estabilidad para los propagadores \mathbf{F} con paso de tiempo δt y \mathbf{G} con paso de tiempo σ , respectivamente, tal que $|r| \leq 1$. Si se cumple que $\frac{\bar{r} - 1}{2} \leq R \leq \frac{\bar{r} + 1}{2}$ con $\bar{r} = r(\lambda \delta t)^s$, entonces el algoritmo parareal es estable para todos los valores posibles de subdominios de $m + 1$ y número de iteraciones.

Demostración. Como ya se ha visto antes, el algoritmo será estable si se cumple $|\bar{r}| \leq 1$ y $|\bar{r} - 2R| \leq 1$. Notar que esta última expresión es una de las suposiciones del teorema por lo que solo haría falta ver que $|\bar{r}| \leq 1$.

Además por las hipótesis del teorema, también se tiene que $|r| \leq 1$ y dado que $\bar{r} = r(\lambda \delta t)^s$, se sigue que $|\bar{r}| = |r|^s \leq 1^s = 1$, cumpliéndose así ambas condiciones. De esta manera se puede asegurar que el algoritmo es estable. \square

3.4. Simulación numérica

Tomamos como ejemplo sencillo el siguiente problema

$$\begin{cases} u'(t) = u(t), & t \in (t_0, T], \\ u(t_0) = u_0, \end{cases} \quad (3.17)$$

y para resolverlo, utilizando el algoritmo parareal, haremos uso del método de Euler explícito tanto como operador grueso como operador fino, con la diferencia de que utilizaremos diferentes pasos de discretización para cada uno.

Definimos:

- $h_f = \frac{T - t_0}{N_t + 1}$, que representa el paso de discretización temporal utilizado para resolver el problema (3.17).
- $h_c = h_f \cdot \text{incr}$, donde incr es un factor de incremento que determinará el paso de discretización temporal del operador fino.

Ahora, podemos presentar una serie de resultados para el siguiente problema en particular

$$\begin{cases} u'(t) = u(t), & t \in (0, 1], \\ u(0) = 0,5. \end{cases} \quad (3.18)$$

Con el objetivo de resolver este problema, hemos implementado el algoritmo parareal en un código de MATLAB, ver código en la subsección A.1 del Anexo A. La solución numérica obtenida se representa gráficamente, mostrando la evolución de la solución a lo largo del tiempo.

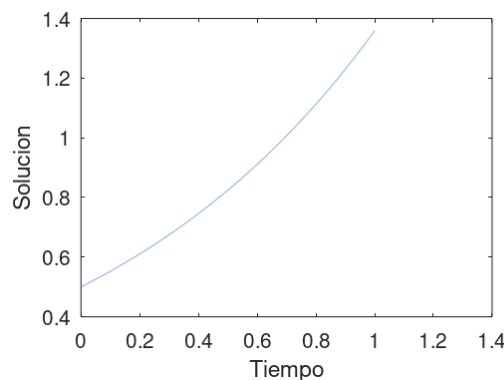


Figura 3.1: Solución numérica del problema (3.18) respecto al tiempo .

En este caso específico, el programa está resolviendo la ecuación diferencial en el intervalo de tiempo $[0, 1]$ con paso de discretización temporal $k = \frac{1}{N_t + 1}$ con $N_t = 20000$. La gráfica muestra la evolución de la solución numérica u a medida que se realiza el proceso iterativo.

Por otro lado, se presenta la siguiente tabla que muestra los resultados que obtenemos al ejecutar el programa para distintos incrementos. En la primera columna, se indica el valor del incremento utilizado, mientras que en la segunda se muestra el número de iteraciones necesarias para alcanzar la convergencia del método.

Incremento	Iteraciones
2	6
4	7
8	8
10	8
16	9

Se observa que el número de iteraciones es independiente al incremento utilizado, ya que el cambio que se produce es mínimo.

Capítulo 4

Aplicación del algoritmo parareal

Una vez vista toda la teoría necesaria, podemos introducir el último capítulo en el que veremos cómo aplicar el algoritmo parareal al modelo SIR básico y al modelo SIR endémico con difusión.

4.1. Parareal para EDOs

En esta sección veremos qué ocurre con el algoritmo parareal aplicado a un sistema de EDOs. Nos basaremos en el modelo SIR básico, cuyo sistema es el siguiente

$$\begin{cases} \frac{dS(t)}{dt} = -\beta S(t)I(t), \\ \frac{dI(t)}{dt} = \beta S(t)I(t) - \nu I(t), \\ \frac{dR(t)}{dt} = \nu I(t), \\ S(0) = S_0 > 0, \quad I(0) = I_0 > 0, \quad R(0) = 0. \end{cases}$$

Hemos diseñado el programa de Matlab incluido en la subsección A.2 del Anexo A con el objetivo de resolver el sistema anterior a través del algoritmo parareal.

Sea $t \in (0, T_{final}]$, consideramos una población con 103 individuos de los cuales, en el instante inicial del estudio, 3 de ellos se encuentran infectados, luego tomamos $S_0 = 100$ e $I_0 = 3$. En cuanto a los parámetros del modelo, definimos la tasa de infección como $\beta = 0,07$ y la tasa de recuperación como $\nu = 0,5$. De esta manera, aplicando el método presentado obtenemos la solución del sistema que representamos en la siguiente gráfica:

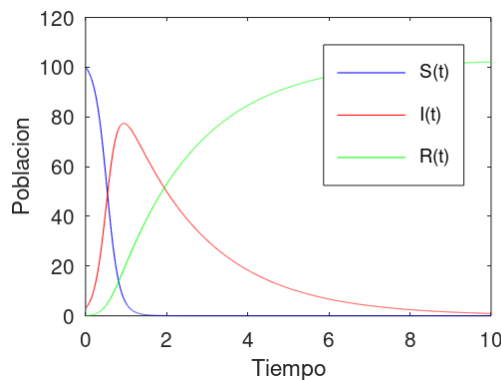


Figura 4.1: Solución numérica a lo largo del tiempo del sistema del modelo SIR para $S_0 = 100$, $I_0 = 3$, $\beta = 0,07$ y $\nu = 0,5$.

Notar que en esta gráfica el número de individuos susceptibles queda representado por la curva azul, la curva roja representa el número de personas infectadas y la curva verde el número de individuos que se recuperan de la enfermedad. Podemos observar como a medida que pasa el tiempo, el número de personas susceptibles disminuye a la vez que el número de infectados aumenta, hasta no quedar ningún individuo que no haya contraído la enfermedad. Además, se observa que el número de personas recuperadas aumenta ligeramente, llegando así a acercarse al número total de la población. Así mismo acaba desapareciendo la enfermedad al no quedar ningún infectado en la población.

Por otro lado, tal y como hemos hecho en el capítulo anterior, mostramos una tabla con los resultados obtenidos al ejecutar el programa para diferentes valores de incremento en el algoritmo parareal.

Incremento	Iteraciones
2	5
4	6
8	6
10	7
16	7

Al igual que en el capítulo anterior, se vuelve a observar que al aumentar el valor del incremento, la variación de las iteraciones es muy pequeña. Por lo que podemos afirmar que el método es robusto con respecto al incremento, ya que el número de iteraciones se mantiene casi constante.

4.2. Parareal para EDPs

Finalmente, veremos qué ocurre con un sistema de EDPsS. Tomamos el sistema correspondiente al modelo SIR endémico con difusión

$$\begin{cases} S_t(x,t) = B(x,t) - \mu S(x,t) - \beta S(x,t)I(x,t) + D_S S_{xx}(x,t), \\ I_t(x,t) = \beta S(x,t)I(x,t) - (\nu + \mu)I(x,t) + D_I I_{xx}(x,t), \\ R_t(x,t) = \nu I(x,t) - \mu R(x,t) + D_R R_{xx}(x,t). \end{cases}$$

Recordamos que ahora las variables dependen del tiempo y espacio, por tanto, sea $n \in \mathbb{N}$, definimos $h = \frac{1}{n+1}$ como el paso de discretización espacial.

Por el sistema (2.12), que se ha obtenido tras el uso del MDF, hemos diseñado el código en Matlab incluido en la subsección A.3 del Anexo A que nos permite resolver el modelo SIR con difusión.

Sea el dominio espacial $[0, 20]$ y $t \in (0, T_{final})$ tomamos los siguientes valores para los parámetros $B = 0$, $\beta = 0,07$, $\nu = 0,5$, $\mu = 0,001$, $D_S = D_I = D_R = 0,01$, $Nt = 10000$, $n = 20$ y representamos los 3 grupos de individuos respecto al espacio para tres distintos valores de T_{final} .

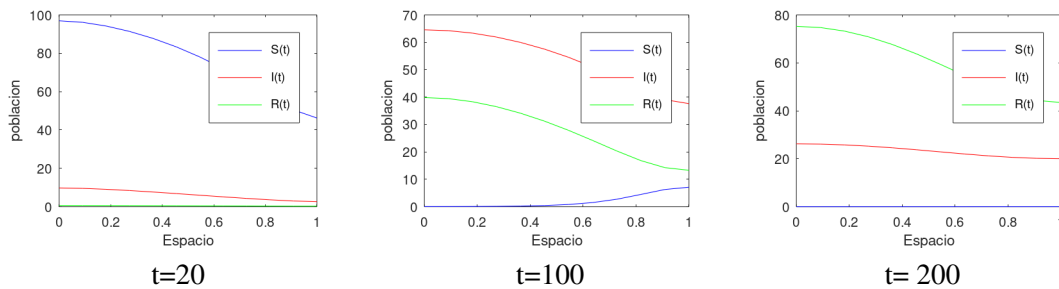


Figura 4.2: Individuos en la población

Notar que, cuando tomamos $t = 100$, el número de infectados supera al número de individuos susceptibles en algún punto de la población, lo que supone que cuanto más pase el tiempo la población entera

acabará infectándose. Este razonamiento se refleja en la tercera gráfica en la que hemos tomado $t = 200$, se observa que ya no hay individuos susceptibles y que el número de individuos va disminuyendo respecto a la gráfica anterior a la vez que aumenta el número de personas recuperadas de la enfermedad.

Por otro lado, presentamos de nuevo la siguiente tabla en la que comparamos las iteraciones del método parareal respecto al valor del incremento

Incremento	Iteraciones
2	5
4	7
8	8
10	9
16	11

Podemos observar, al igual que en los casos anteriores, que la variación de las iteraciones es muy pequeña al aumentar el incremento, lo que indica que el método vuelve a ser robusto con respecto al incremento.

Bibliografía

- [1] LAURA GARCÍA ROVIRA. *Modelos matemáticos compartimentales en epidemiología* <https://riull.ull.es/xmlui/bitstream/handle/915/6779/Modelos%20matematicos%20compartimentales%20en%20epidemiologia.pdf?sequence=1&isAllowed=y>
- [2] ANDREA GARCÍA PIÑERA. *Modelos de ecuaciones diferenciales para la propagación de enfermedades infecciosas*. <https://repositorio.unican.es/xmlui/bitstream/handle/10902/7125/Andrea%20Garcia%20Pi%C3%B1era.pdf>
- [3] FRED BRAUER, CARLOS CASTILLO-CHÁVEZ, ELMER DE LA PAVA-SALGADO, KAMAL BARLEY, CARLOS W. CASTILLO-GARSOW, DIEGO CHOWELL BALTAZAR ESPINOZA, PAULA GONZÁLEZ PARRA, CARLOS HERNÁNDEZ SUÁREZ Y VÍCTOR M. MORENO. *Modelos de la propagación de enfermedades infecciosas*. Cali. Universidad Autónoma de Occidente (2015).
- [4] VARSHA TOLANI BHAGTANI. *Generalización del modelo SIR para la predicción de propagación de epidemias*. https://oa.upm.es/71391/1/TFM_VARSHA_TOLANI_BHAGTANI.pdf
- [5] *Publicació electrònica de divulgació del Departament de Matemàtiques de la Universitat Autònoma de Barcelona*. Volum 2013, treball no. 3, 17 pp. ISSN: 1887-1097. <https://mat.uab.cat/~matmat/PDFv2013/v2013n03.pdf>
- [6] FRANCISCO JOSÉ GASPAR. *Apuntes de la asignatura. “Ecuaciones diferenciales parciales”, Universidad de Zaragoza*.
- [7] LUIS LARA ROMERO, ZENNER CHÁVEZ ALIAGA Y JOSÉ CASTAÑEDA VERGARA. *El método de diferencias finitas. Teoría y práctica*, Universidad privada antenor orrego. 135-187, (2019).
- [8] LUIS FERRAGUT CANALS. *Análisis Numérico del Método de Diferencias Finitas para Ecuaciones en Derivadas Parciales*. <https://gredos.usal.es/bitstream/handle/10366/145828/MetodoDiferenciasFinitas.pdf?sequence=1&isAllowed=y>
- [9] RANDALL J. LEVEQUE. *Finite Difference Methods for Ordinary and Partial Differential Equations*. SIAM (1955).
- [10] IÑIGO JIMENEZ CIGA. *Parallel in Time and Space Methods for the Numerical Solution of Evolutionary Reaction-Diffusion Problems*
- [11] NOELIA ORTEGA ROMÁN. *Métodos paralelos en tiempo para ecuaciones en derivadas parciales*. <https://rodin.uca.es/handle/10498/24720>

A. Códigos

A.1 Parareal

```
1
2 %parareal del problema  $u'(t)=u(t)$  con F y G Euler explícito
3
4 tinicial = 0;
5 tfinal = 10;
6 u0 = 0.5;
7
8 Nt=10000
9 hf=(tfinal-tinicial)/Nt;
10
11 incr=20% razon entre tama o de pasos de parareal
12 hc=incr*hf;
13 t=linspace(tinicial,tfinal,Nt+1);
14
15 % Inicializamos los tama os de las soluciones
16 uf=zeros(Nt+1,1); % soluci n fina
17 uc=zeros(Nt/incr+1,1); % soluci n basta
18
19 % Condici n inicial
20 uc(1)=u0;
21 uf(1)=uc(1);
22
23 for i=1:Nt/incr
24     uc(i+1)=uc(i)*(1+hc);
25     t=t+hc;
26 end
27
28 it=1;
29 itmax=200;
30 error=1;
31 tol=1e-8;
32 uc_old=uc;
33 %itera hasta que se alcanza el n mero maximo de iteraciones o se alcanza una
    tolerancia de error tol
34 while it<itmax && error>tol
35     for i=1:Nt/incr
36         uf((i-1)*incr+1)=uc_old(i);
37         u2=uc(i)*(1+hc); %G(i+1)
38         u3=uc_old(i)*(1+hc);%G(i)
39         for j=1:incr
40             uf((i-1)*incr+j+1)=uf((i-1)*incr+j)*(1+hf);
41
42         end
43         uc(i+1)=uf(i*incr+1)+u2-u3;
44     end
45     error(it)=max(abs(uc-uc_old))
46     uc_old=uc;
47     it=it+1;
```

```

48
49 end
50 plot(t, uf)
51 xlabel('Tiempo');
52 ylabel('Solucion');
53 %semilogy(error)
54 %xlabel('iteraciones');
55 %ylabel('error');
56 %error
57 uc
58 it

```

A.2 Parareal para el modelo SIR

```

1 %parareal para las edos del MODELO SIR
2
3 beta= 0.07
4 nu=0.5
5 s0= 100
6 i0= 3
7 r0= 0
8
9 tinicial = 0;
10 tfinal = 10;
11
12 Nt=1000;
13 hf=(tfinal-tinicial)/Nt;
14
15 incr=2 % razon entre tama o de pasos de parareal
16 hc=incr*hf;
17 t=linspace(tinicial,tfinal,Nt+1);
18
19 % Inicializamos los tama?os de las soluciones
20 uc_S = zeros(Nt/incr+1, 1);
21 uc_I = zeros(Nt/incr+1, 1);
22 uc_R = zeros(Nt/incr+1, 1);
23
24 uf_S = zeros(Nt+1, 1);
25 uf_I = zeros(Nt+1, 1);
26 uf_R = zeros(Nt+1, 1); % Matriz para la soluci n fina (uf)
27 % Condici?n inicial
28 uc_S(1) = s0;
29 uc_I(1) = i0;
30 uc_R(1) = r0;
31
32 uf_S(1) = uc_S(1);
33 uf_I(1) = uc_I(1);
34 uf_R(1) = uc_R(1);
35
36
37 for i=1: Nt/incr
38     uc_S(i+1) = uc_S(i) * (1 - beta*hc*uc_I(i));
39     uc_I(i+1) = uc_I(i) * (1 + beta*hc * uc_S(i) - nu*hc);
40     uc_R(i+1) = uc_R(i) + nu*hc*uc_I(i);
41 endfor
42
43 it=1;
44 itmax=200;
45 error=1;
46 tol=1e-8;
47
48 uc_old_S= uc_S;
49 uc_old_I= uc_I;

```

```

50 uc_old_R= uc_R;
51
52 while it < itmax && error > tol
53
54     for i = 1:Nt/incr
55         uf_S((i-1)*incr+1) = uc_old_S(i);
56         uf_I((i-1)*incr+1) = uc_old_I(i);
57         uf_R((i-1)*incr+1) = uc_old_R(i);
58
59         u2_S = uc_S(i) * (1 - beta*hc*uc_I(i));
60         u2_I = uc_I(i) * (1 + beta*hc * uc_S(i) - nu*hc);
61         u2_R = uc_R(i) + nu*hc*uc_I(i);
62
63         u3_S = uc_old_S(i) * (1 - beta*hc*uc_old_I(i));
64         u3_I = uc_old_I(i) * (1 + beta*hc * uc_old_S(i) - nu*hc);
65         u3_R = uc_old_R(i) + nu*hc*uc_old_I(i);
66
67         for j = 1:incr
68             uf_S((i-1)*incr+j+1) = uf_S((i-1)*incr+j) * (1 - beta*hf*uf_I((i-1)*
69             incr+j));
70             uf_I((i-1)*incr+j+1) = uf_I((i-1)*incr+j) * (1 + beta*hf * uf_S((i
71             -1)*incr+j) - nu*hf);
72             uf_R((i-1)*incr+j+1) = uf_R((i-1)*incr+j) + nu*hf*uf_I((i-1)*incr+j)
73             ;
74             t = t + hf;
75         end
76         uc_S(i+1) = uf_S(i*incr+1) + u2_S - u3_S;
77         uc_I(i+1) = uf_I(i*incr+1) + u2_I - u3_I;
78         uc_R(i+1) = uf_R(i*incr+1) + u2_R - u3_R;
79     end
80     %si no defino este error no me parara nunca el bucle de antes , solo parara
81     %por las it max
82     %criterio de parada
83     error(it)=max(abs(uc_S-uc_old_S))+max(abs(uc_I-uc_old_I))+ max(abs(uc_R-
84     uc_old_R));
85     uc_old_S = uc_S;
86     uc_old_I = uc_I;
87     uc_old_R = uc_R;
88 end
89 plot(t, uf_S, 'b', t, uf_I, 'r', t, uf_R, 'g')
90 xlabel('Tiempo');
91 ylabel('Poblacion');
92 semilogy(error)
93 xlabel('iteraciones');
94 ylabel('error');
95
96 uc_S
97 uc_I
98 uc_R
99 %error
100 it

```

A.3 Parareal para el modelo SIR endémico con difusión

```

1 %parareal para las edos del MODELO SIR ENDEMICO CON DIFUSION
2 %PARAMETROS
3 beta= 0.07;
4 nu=0.5;
5 mu=0.001;
6 Ds=0.01;
7 Di=0.01;
8 Dr=0.01;

```

```

9  tinicial=0;
10 tfinal= 10
11 n=2;
12 Nt=10000 % incr*N
13
14 t=linspace(tinicial,tfinal,Nt+2);
15 kf=(tfinal-tinicial)/Nt;
16 incr=8 % razon entre tama o de pasos de parareal
17 kc=incr*kf;
18
19 h=1/(n+1);
20 x=0:h:1;
21
22 rc = kc/h^2;
23 rf = kf/h^2;
24
25 s_c = zeros(n+2,Nt/incr+1);
26 i_c = zeros(n+2,Nt/incr+1);
27 r_c= zeros(n+2,Nt/incr+1);
28
29 s_f = zeros(n+2, Nt+1);
30 i_f = zeros(n+2, Nt+1);
31 r_f = zeros(n+2, Nt+1);
32
33 s_2 = zeros(n+1,1);
34 s_3 = zeros(n+1,1);
35 r_2 = zeros(n+1,1);
36 r_3 = zeros(n+1,1);
37 i_2 = zeros(n+1,1);
38 i_3 = zeros(n+1,1);
39 %condiciones iniciales
40 PopInicial=4.5*((x.^3)-6*(x.^2)-x+10);
41 S=sum(PopInicial);
42 PopInicial=1000/S*PopInicial;
43 i_c(:, 1) = ones(n+2,1);
44 s_c(:, 1) = PopInicial'-i_c(:, 1);
45 r_c(:, 1) = zeros(n+2,1) ;
46
47 s_f(:,1)=s_c(:,1);
48 i_f(:,1)=i_c(:,1);
49 r_f(:,1)=r_c(:,1);
50
51
52 for j=1: Nt/incr
53     s_c(1,j+1)=(1-mu*kc-2*Dc*rc)*s_c(1,j) + 2*Dc*rc*s_c(2,j) - kc*beta*s_c(1,j)
54     *i_c(1,j);
55     i_c(1,j+1) =(1-(mu + nu)*kc-2*Di*rc)*i_c(1,j)+ 2*Di*rc*i_c(2,j) + kc*beta*
56     s_c(1,j)*i_c(1,j);
57     r_c(1,j+1) = (1-mu*kc-2*Dr*rc)*r_c(1,j)+ 2*Dr*rc*r_c(2,j) +kc*nu*i_c(1,j);
58     for i=2:n+1
59         s_c(i,j+1) =(1-mu*kc-2*Dc*rc)*s_c(i,j) + Dc*rc*s_c(i-1,j) + Dc*rc*
60         s_c(i+1,j) - kc*beta*s_c(i,j)*i_c(i,j);
61         i_c(i,j+1) = (1-(mu + nu)*kc-2*Di*rc)*i_c(i,j)+Di*rc*i_c(i-1,j) + Di
62         *rc*i_c(i+1,j) + kc*beta*s_c(i,j)*i_c(i,j);
63         r_c(i,j+1) = (1-mu*kc-2*Dr*rc)*r_c(i,j)+ Dr*rc*r_c(i-1,j)+ Dr*rc*
64         r_c(i+1,j) + kc*nu*i_c(i,j);
65     end
66     s_c(n+2,j+1) =(1-mu*kc-2*Dc*rc)*s_c(n+2,j) + 2*Dc*rc*s_c(n+1,j) - kc*beta*
67     s_c(n+2,j)*i_c(n+2,j);
68     i_c(n+2,j+1) =(1-(mu + nu)*kc-2*Di*rc)*i_c(n+2,j)+ 2*Di*rc*i_c(n+1,j) + kc
69     *beta*s_c(n+2,j)*i_c(n+2,j);
70     r_c(n+2,j+1) = (1-mu*kc-2*Dr*rc)*r_c(n+2,j) + 2*Dr*rc*r_c(n+1,j)+kc*nu*i_c
71     (n+2,j);

```

```

64     end
65
66     it=1;
67     itmax=200;
68     error=1;
69     tol=1e-8;
70
71     sold_c=s_c;
72     iold_c=i_c;
73     rold_c=r_c;
74
75     while it < itmax & error > tol
76
77         for j = 1:Nt/incr
78             s_f(:,(j-1)*incr+1) = sold_c(:,j);
79             i_f(:,(j-1)*incr+1) = iold_c(:,j);
80             r_f(:,(j-1)*incr+1) = rold_c(:,j);
81
82             s_2(1)=(1-mu*kc-2*D*rc)*s_c(1,j) + 2*D*rc*s_c(2,j) - kc*beta*s_c
(1,j)*i_c(1,j);
83             i_2(1)=(1-(mu + nu)*kc-2*Di*rc)*i_c(1,j)+ 2*Di*rc*i_c(2,j) + kc*
beta*s_c(1,j)*i_c(1,j);
84             r_2(1) = (1-mu*kc-2*Dr*rc)*r_c(1,j)+ 2*Dr*rc*r_c(2,j) +kc*nu*i_c(1,
j);
85             for i=2:n+1
86                 s_2(i) =(1-mu*kc-2*D*rc)*s_c(i,j) + D*rc*s_c(i-1,j) + D*rc*
s_c(i+1,j) - kc*beta*s_c(i,j)*i_c(i,j);
87                 i_2(i) = (1-(mu + nu)*kc-2*Di*rc)*i_c(i,j)+Di*rc*i_c(i-1,j) + Di
*rc*i_c(i+1,j) + kc*beta*s_c(i,j)*i_c(i,j);
88                 r_2(i) = (1-mu*kc-2*Dr*rc)*r_c(i,j)+ Dr*rc*r_c(i-1,j)+ Dr*rc*
r_c(i+1,j) + kc*nu*i_c(i,j);
89             end
90             s_2(n+2) =(1-mu*kc-2*D*rc)*s_c(n+2,j) + 2*D*rc*s_c(n+1,j) - kc*
beta*s_c(n+2,j)*i_c(n+2,j);
91             i_2(n+2) =(1-(mu + nu)*kc-2*Di*rc)*i_c(n+2,j)+ 2*Di*rc*i_c(n+1,j)
+ kc*beta*s_c(n+2,j)*i_c(n+2,j);
92             r_2(n+2) = (1-mu*kc-2*Dr*rc)*r_c(n+2,j) + 2*Dr*rc*r_c(n+1,j)+kc*nu
*i_c(n+2,j);
93
94             %definimos s3
95             s_3(1)=(1-mu*kc-2*D*rc)*sold_c(1,j) + 2*D*rc*sold_c(2,j) - kc*
beta*sold_c(1,j)*iold_c(1,j);
96             i_3(1) =(1-(mu + nu)*kc-2*Di*rc)*iold_c(1,j)+ 2*Di*rc*iold_c(2,j) +
kc*beta*sold_c(1,j)*iold_c(1,j);
97             r_3(1) = (1-mu*kc-2*Dr*rc)*rold_c(1,j)+ 2*Dr*rc*rold_c(2,j) +kc*nu*
iold_c(1,j);
98             for i=2:n+1
99                 s_3(i) =(1-mu*kc-2*D*rc)*sold_c(i,j) + D*rc*sold_c(i-1,j) + D*rc*
sold_c(i+1,j) - kc*beta*sold_c(i,j)*iold_c(i,j);
100                 i_3(i) = (1-(mu + nu)*kc-2*Di*rc)*iold_c(i,j)+Di*rc*iold_c(i-1,j)
+ Di*rc*iold_c(i+1,j) + kc*beta*sold_c(i,j)*iold_c(i,j);
101                 r_3(i) = (1-mu*kc-2*Dr*rc)*rold_c(i,j)+ Dr*rc*rold_c(i-1,j)+ Dr
*rc*rold_c(i+1,j) + kc*nu*iold_c(i,j);
102             end
103             s_3(n+2) =(1-mu*kc-2*D*rc)*sold_c(n+2,j) + 2*D*rc*sold_c(n+1,j)
- kc*beta*sold_c(n+2,j)*iold_c(n+2,j);
104             i_3(n+2) =(1-(mu + nu)*kc-2*Di*rc)*iold_c(n+2,j)+ 2*Di*rc*iold_c(n
+1,j) + kc*beta*sold_c(n+2,j)*iold_c(n+2,j);
105             r_3(n+2) = (1-mu*kc-2*Dr*rc)*rold_c(n+2,j) + 2*Dr*rc*rold_c(n+1,j)
+kc*nu*iold_c(n+2,j);
106
107
108             for l = 1:incr

```

```

109         s_f(1,(j-1)*incr+ 1+1)=(1-mu*kf-2*Dsr*f)*s_f(1,(j-1)*incr+ 1) +
        2*Dsr*f*s_f(2,(j-1)*incr+ 1) - kf*beta*s_f(1,(j-1)*incr+ 1)*i_f(1,(j-1)*
110         incr+ 1);
        i_f(1,(j-1)*incr+ 1+1) =(1-(mu + nu)*kf-2*Di*r*f)*i_f(1,(j-1)*
        incr+ 1)+ 2*Di*r*f*i_f(2,(j-1)*incr+ 1) + kf*beta*s_f(1,(j-1)*incr+ 1)*i_f
        (1,(j-1)*incr+ 1);
111         r_f(1,(j-1)*incr+ 1+1) = (1-mu*kf-2*Dr*r*f)*r_f(1,(j-1)*incr+ 1)
        + 2*Dr*r*f*r_f(2,(j-1)*incr+ 1) +kf*nu*i_f(1,(j-1)*incr+ 1);
112
113         for i=2:n+1
114             s_f(i,(j-1)*incr+ 1+1) =(1-mu*kf-2*Dsr*f)*s_f(i,(j-1)*incr+
        1) + Dsr*f*s_f(i-1,(j-1)*incr+ 1) + Dsr*f*s_f(i+1,(j-1)*incr+ 1) - kf*beta*
        s_f(i,(j-1)*incr+ 1)*i_f(i,(j-1)*incr+ 1);
115             i_f(i,(j-1)*incr+ 1+1) = (1-(mu + nu)*kf-2*Di*r*f)*i_f(i,(j
        -1)*incr+ 1)+Di*r*f*i_f(i-1,(j-1)*incr+ 1) + Di*r*f*i_f(i+1,(j-1)*incr+ 1) +
        kf*beta*s_f(i,(j-1)*incr+ 1)*i_f(i,(j-1)*incr+ 1);
116             r_f(i,(j-1)*incr+ 1+1) = (1-mu*kf-2*Dr*r*f)*r_f(i,(j-1)*incr+
        1)+ Dr*r*f*r_f(i-1,(j-1)*incr+ 1)+ Dr*r*f*r_f(i+1,(j-1)*incr+ 1) + kf*nu*i_f
        (i,(j-1)*incr+ 1);
117         end
118         s_f(n+2,(j-1)*incr+ 1+1) =(1-mu*kf-2*Dsr*f)*s_f(n+2,(j-1)*incr
        + 1) + 2*Dsr*f*s_f(n+1,(j-1)*incr+ 1) - kf*beta*s_f(n+2,(j-1)*incr+ 1)*i_f(n
        +2,(j-1)*incr+ 1);
119         i_f(n+2,(j-1)*incr+ 1+1) =(1-(mu + nu)*kf-2*Di*r*f)*i_f(n+2,(j
        -1)*incr+ 1)+ 2*Di*r*f*i_f(n+1,(j-1)*incr+ 1) + kf*beta*s_f(n+2,(j-1)*incr+ 1
        )*i_f(n+2,(j-1)*incr+ 1);
120         r_f(n+2,(j-1)*incr+ 1+1) = (1-mu*kf-2*Dr*r*f)*r_f(n+2,(j-1)*
        incr+ 1) + 2*Dr*r*f*r_f(n+1,(j-1)*incr+ 1)+kf*nu*i_f(n+2,(j-1)*incr+ 1);
121
122         end
123         s_c(:,j+1) = s_f(:,j*incr+1) + s_2 - s_3;
124         i_c(:,j+1) = i_f(:,j*incr+1) + i_2 - i_3;
125         r_c(:,j+1) = r_f(:,j*incr+1) + r_2 - r_3;
126     end
127     error(it) = max(max(abs(s_c-sold_c)))+ max(max(abs(i_c-iold_c)))
        + max(max(abs(r_c-rold_c)));
128     sold_c=s_c;
129     iold_c=i_c;
130     rold_c=r_c;
131     it = it + 1;
132 end
133
134 semilogy(error)
135 xlabel('Iteraciones');
136 ylabel('error');
137
138 %s_c
139 %i_c
140 %r_c
141 it

```