Universidade Estadual de Maringá
Centro de Ciências Exatas
Departamento de Física

Trabalho de Conclusão de Curso

Estudando a função de autocorrelação no jogo Pedra Papel e Tesoura

Acadêmico: José Vítor de Oliveira Silva

Orientador: Prof. Dr. Breno Ferraz de Oliveira

Maringá, 25 de fevereiro de 2016

Universidade Estadual de Maringá
Centro de Ciências Exatas
Departamento de Física

Trabalho de Conclusão de Curso

Estudando a função de autocorrelação no jogo Pedra Papel e Tesoura

Trabalho de Conclusão de Curso apresentado ao Departamento de Física da Universidade Estadual de Maringá, sob orientação do professor Dr. Breno Ferraz de Oliveira, como parte dos requisitos para obtenção do título de bacharel em Física.

Acadêmico: José Vítor de Oliveira Silva

Orientador: Prof. Dr. Breno Ferraz de Oliveira

Maringá, 25 de fevereiro de 2016

Sumário

Agradecimentos

Gostaria de agradecer ao meu orientador Dr. Breno Ferraz de Oliveira pela orientação e pela paciência e dedicação transmitida ao meu trabalho. Aos meus colegas de graduação, pelo companheirismo durante o curso. E por último e não menos importante gostaria de agradece aos meus familiares por todo o apoio dado.

Resumo

No decorrer deste trabalho foi estudado a dinâmica de população com três espécies que interagem entre si conforme as regras do jogo pedra tesoura e papel, ou seja, competindo de maneira cíclica. No decorrer do estudo foram utilizados dois modelos para descrever esta interação, o modelo Estocástico e o modelo de May-Leonard. Foi observado padrões espirais com três braços e que a evolução desses padrões no tempo levam a espirais com larguras características L com valor constante, assegurando dessa forma a existência de mais de três espécies na rede. Com objetivo de quantizar L foi calculado a função de autocorrelação.

Introdução

No ano de 1996 foi realizado um trabalho onde foi observado uma competição do tipo RPS (Rock Paper Scissors) [1]. Durante o trabalho foram estudados padrões de comportamento de machos da espécies e uso do território da espécie de lagarto *Uta stansburiana*. A forma com a qual os machos defendem seu território é determinada de maneira genética e pode ser identificada a partir do padrão de cores da garganta dos machos. Nos machos com garganta alaranjada é observado um comportamento mais agressivo e grandes territórios. Em machos com garganta azul é observado um comportamento menos agressivo em relação aos de garganta laranja, e com territórios menores. Os machos de garganta amarela não possuem nenhum comportamento territorial, ou seja eles não tem um território somente deles, para reprodução eles usam de furtividade uma vez que sua coloração é parecida com a das fêmeas.

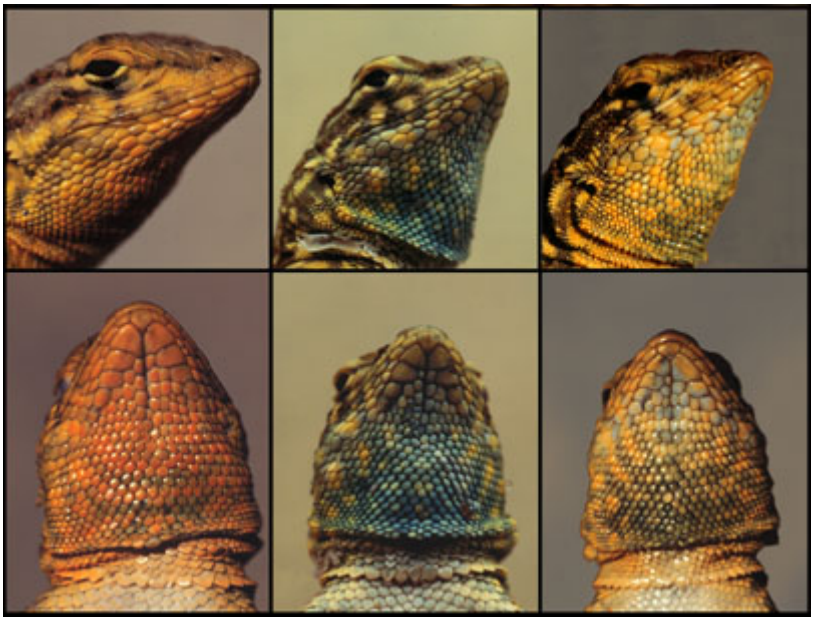


Figura 1: Padrões de coloração da espécie de lagarto *Uta stansburiana*

Como o lagarto de coloração laranja tem uma área muito grande para vigiar fica difícil vigiar todos os fêmeas do território, o lagarto amarelo se aproveita desta situação e se infiltra no meio das fêmeas para se reproduzir . O lagarto Laranja por mais agressivo conquista os territórios do lagarto azul e assim fica com as fêmeas do território. O lagarto azul por sua vez por possuir um território menor, faz com que ele tenha mais controle sobre todos as fêmeas e assim possa detectar os lagartos amarelos invasores. As relações de acasalamento observadas entre os três lagartos se comportam de maneira , por ser hereditária ,gera um ciclo de seis anos de abundancia das estratégias, fazendo com que se mantenha a diversidade do sistema.

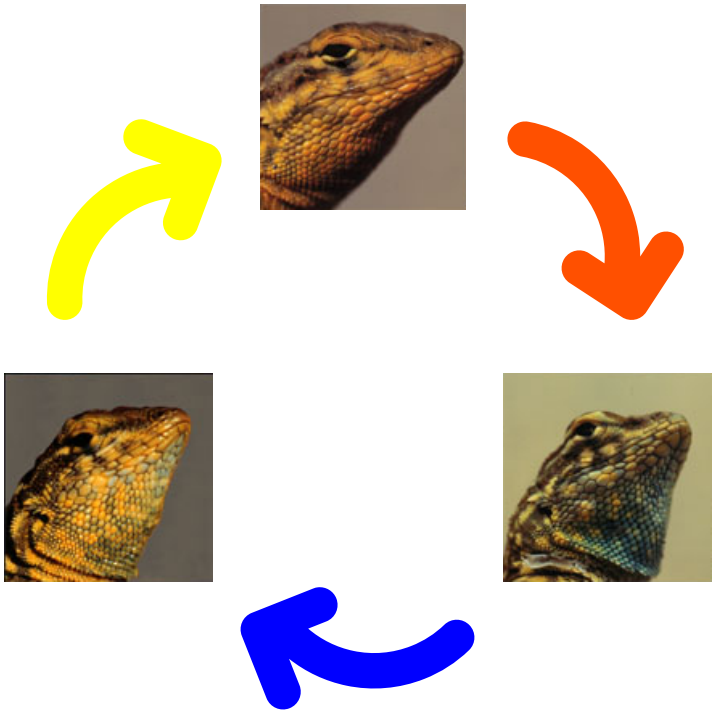


Figura 2: Interação cíclica entre os lagartos da espécies *Uta stansburiana*

No ano de 2002 foi publicado na revista Nature [2] um experimento realizado com bactérias, onde foi mostrado que as regras do jogo RPS podem ser utilizadas para explicar o padrão formado pelas colonias dessas bactérias. Posteriormente em [3] os autores mostram por meio de simulações que a introdução da mobilidade pode alterar de maneira significativa os esses padrões. Uma forma de quantificar esses padrões consiste em calcular a função de autocorrelação.

A função de autocorrelação é uma ferramenta matemática muito importante para estudar a evolução espacial de uma rede. Essa pode ser encontrada após o cálculo da seguinte integral (recomendamos [4] para mais detalhes):

$$C(\vec{r}) = \int_{\text{toda rede}} \phi(\vec{r})\phi(\vec{r}+\vec{r'})d\vec{r'} \ .$$

onde $\phi(\vec{r})$ é um campo escalar que descreve o valor de alguma grandeza física em todos os pontos da rede. Apesar das informações que a função de autocorrelação pode fornecer, há um custo computacional tão elevado que pode inviabilizar seu cálculo. Uma forma de contornar esse problema consiste em calcular a função de autocorrelação no espaço de Fourier, isto é,

$$S(\vec{k}) = \phi(\vec{k})\phi^*(\vec{k}) \ ,$$

onde $\phi^*(\vec{k})$ é o complexo conjugado do campo $\phi(\vec{k})$ no espaço de Fourier. Note que agora não há mais a dependência da integral. A transformada inversa da função $S(\vec{k})$ fornece a função de autocorrelação $C(\vec{r})$ [4]. A transformada do campo $\phi(\vec{r})$ e a transformada inversa da função $S(\vec{k})$ podem ser feitas utilizando a biblioteca FFTW (*Fastest Fourier Transform in the West*).

Durante este trabalho de conclusão de curso iremos simular estas interações utilizando simulações estocásticas e soluções de equações diferenciais. Para tal propósito descreveremos alguns métodos computacionais que serão utilizados neste trabalho.

No capitulo 1 serão apresentados métodos de cálculo numérico, estes serão implementados durante a escrita do programa que ira fazer as simulações usando o modelo de equações diferenciais. No capítulo 2 serão apresentadas as regras que do jogo pedra papel e tesoura que será adotada durante as simulações. No capítulo 3 serão apresentados os resultados e feitas considerações sobre os resultados das simulações. Os códigos usados durante as simulações se encontram nos apêndices.

Capítulo 1

Métodos Computacionais

Neste capítulo serão apresentados métodos de cálculo numérico. Estes métodos serão utilizados para escrever o código computacional que irá realizar as simulações com equações diferenciais.

1.1 Derivada Numérica

1.1.1 Derivada primeira

Matematicamente a definição de derivada para uma função $f(x)$ é dada por

$$\frac{d}{dx}f(x) = f'(x) = \lim_{h\to 0}\frac{f(x+h)-f(x)}{h}, \tag{1.1}$$

onde h é um passo da função $f(x)$ [5]. Como no computador não é possível fazer $h \to 0$ é necessário encontrar uma expressão que leva em conta o tamanho h. A fim de encontrar uma expressão para a derivada da $f(x)$ faremos a expansão de $f(x)$ em série de Taylor em torno de um ponto arbitrário que chamaremos de ponto a

$$f(x) = f(a)(x-a)^0 + f'(a)(x-a)^1 + \frac{(x-a)^2 f''(a)}{2} + ... \tag{1.2}$$

Como estamos procurando uma expressão para $f'(x)$ e não para $f'(a)$ iremos fazer a expansão da função $f(x+h)$ em torno do ponto $a = x$.

$$f(x+h) = f(x) + hf'(x) + \frac{h^2 f''(x)}{2} + ... \ .$$

Usando os dois primeiros termos é possível estimar a derivada como

$$f'(x) \approx \frac{f(x+h)-f(x)}{h},$$

logo

$$f'(x) = \frac{f(x+h)-f(x)}{h} + \mathcal{O}(h). \tag{1.3}$$

O termo $\mathcal{O}(h)$ representa o erro da derivada. No caso a ordem do erro da $f'(x)$ é da ordem h.

Também é possível definir a derivada de outra forma, para tal feito iremos usar expansão em série de Taylor da equação(1.2), mas agora com $f(x) = f(x-h)$, e $a = x$, assim obtemos a seguinte expressão

$$f'(x-h) = f(x) - hf'(x) + \frac{h^2 f''(x)}{2}....,$$

Novamente tomando os dois primeiros termos é possível estimar a derivada como

$$f'(x) \approx \frac{f(x) - f(x-h)}{h} \approx f'(x) + \frac{h^2 f''(x)}{2}...,$$

ou ainda,

$$f'(x) = \frac{f(x) - f(x-h)}{h} + \mathcal{O}(h). \tag{1.4}$$

Suponha agora que a f seja definida por $f(x) = a + bx^2$. Temos que a sua derivada aproximada é igual a

$$f' = 2bx + bh \ ,$$

uma vez que sabemos o valor exato da derivada ($f' = 2bx$) é possível notar que a resposta é a mesma a menos de um termo bh, esse termo corresponde ao erro da derivada numérica. No entanto se b não é muito grande e se tomarmos um h pequeno teremos um resultado próximo ao exato. Porém não podemos tomar o valor de h muito pequeno, pois a subtração $f(x+h) - f(x)$ pode gerar alguns arredondamentos e levando assim a uma perda de precisão.

Para que a nossa aproximação seja mais acurada precisamos adicionar mais termos da série de Taylor a nossa expressão. Dessa forma,

$$f(x \pm h) = f(x) \pm hf' + \frac{h^2 f''}{2}.$$

Subtraindo $f(x+h)$ de $f(x-h)$ e reagrupando os termos

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2 f'''}{6} + ..., \tag{1.5}$$

é possível observar que o termo dominante no erro possui potência de h^2, se nós truncarmos na derivada terceira.

1.1.2 Derivada Segunda

Para definir a derivada segunda basta tomar a derivada da derivada

$$f''(x) = \frac{f'(x+h) - f'(x-h)}{2h},$$
$$f'(x) = \frac{f(x+h) - f(x-h)}{2h},$$

substituindo f' em f'' temos

$$f''(x) = \frac{f(x+h+h) - f(x-h+h) - f(x-h+h) - f(x-h-h)}{2h2h},$$

reescrevendo

$$f''(x) = \frac{f(x+2h) - 2f(x) - f(x-2h)}{4h},$$

fazendo agora $2h$ como H e reescrevendo temos

$$f''(x) = \frac{f(x+H) - 2f(x) - f(x-H)}{H^2} + \mathcal{O}(H^2), \tag{1.6}$$

o erro neste caso, é da ordem de H^2. Então agora surge uma dúvida "Qual é o melhor valor de H?".

1.2 Erro Analitico

Para mostrar a dependência de h no cálculo da derivada foi utilizada a derivada segunda da função e^x e diferentes valores de h. Como medida de erro, retiramos de [5] a seguinte expressão:

$$\epsilon = log_{10}\left(\frac{f''_{computador} - f''_{exato}}{f''_{exato}}\right), \tag{1.7}$$

onde $f''_{computador}$ é a derivada calculada de maneira numérica e f''_{exato} é o valor da derivada calculada de maneira analítica.

A fim de apresentar os dados obtidos de maneira simples foi confeccionado gráfico de ϵ por h.

Pela Figura 1.1 é possível notar que o melhor valor h para a derivada segunda de e^x é da ordem de 10^{-4}. É importante ressaltar que esse valor de h vale para e^x, nos outros casos, deve haver um outros valores de h que minimizem os erro. Respondendo a pergunta feita sobre qual o melhor valor de h, a melhor forma de se encontrar o valor de h é calcular o valor da derivada numérica e comparar com o valor da derivada calculada de maneira analítica .

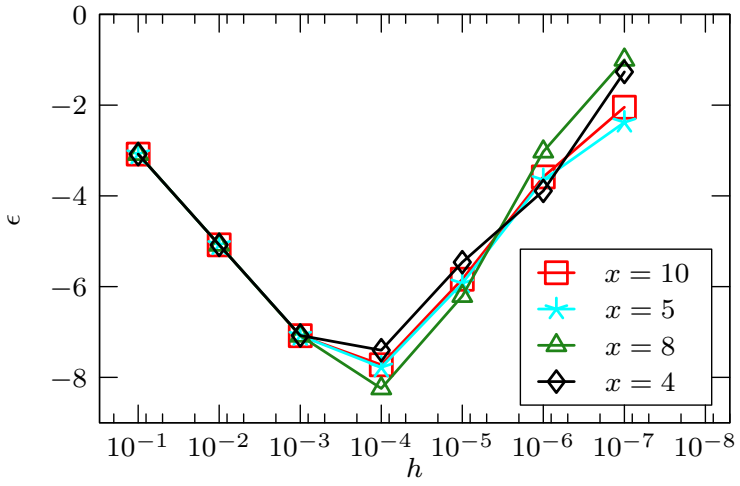


Figura 1.1: Erro analítico.

1.3 Transformada de Fourier

Outro método essencial para nosso trabalho é aplicar a transformada de Fourier nas simulações. Para isso, é necessário introduzir conceitos como integral numérica que será abordada na próxima secção.

1.3.1 Integral Númerica

A integral possui um significado simples, se considerar a figura 1.2 a expressão (1.8) representa a areá S abaixo da curva começando em no ponto $x = a$ e terminando no ponto $x = b$.

$$I_{ab} = \int_a^b f(x)dx = \lim_{\Delta x_i \to 0} \sum_{i=1}^{N} f(x_i)\Delta x_i. \tag{1.8}$$

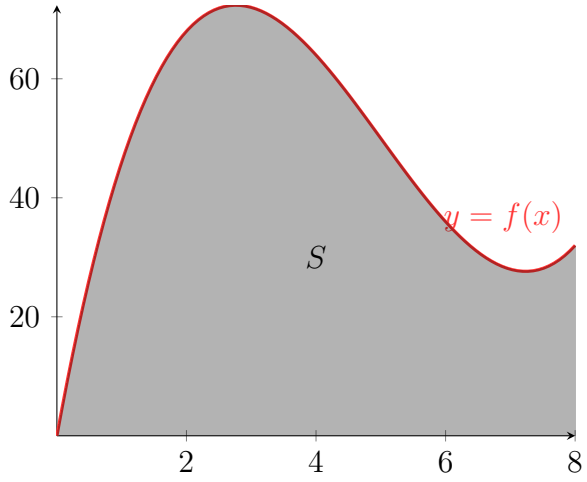


Figura 1.2: Representação da integral calculada do ponto a até o ponto b.

A integração numérica é usada quando não se sabe a expressão da $f(x)$, neste caso a somatória (1.8) não é calculada no limite de $\Delta x_i \to 0$, mas em Δx_i pequenos. Para o caso de funções bem comportadas é possível usar $\Delta x_i = \Delta x$, ou seja todos os valores iguais. Assim a equação (1.8) fica da seguinte forma

$$I_{ab} = \Delta x \sum_{i=1}^{n} f(x_i) = \Delta x \sum_{i=1}^{n} f_i. \tag{1.9}$$

Geometricamente calcular a integral dessa forma implica em construir um retângulo entre x_i e x_{i+1}, de altura f_i, a cada x_i e aproximar a área entre a curva $f(x)$ e o eixo x como a soma da área dos retângulos conforme mostra a figura 1.3.

E possível melhorar a aproximação da integral numérica se usarmos ao invés de retângulos, trapézios conforme mostra a figura 1.4.

Isto corresponde a fazer-se uma interpolação linear entre os pontos (x_i, f_i) consecutivos. A área do trapézio corresponde $(\frac{1}{2}(f_i + f_{i+1})\Delta x)$.

Somando-se as áreas dos trapézios consecutivos temos

$$I_{ab} = \Delta x(\frac{1}{2}f_1 + f_2 + f_3 + ... + f_{n-1} + \frac{1}{2}f_N)I_{ab} = \Delta x \sum_{i=1}^{N}(f_i - \frac{1}{2}(f_1 + f_N)), \tag{1.10}$$

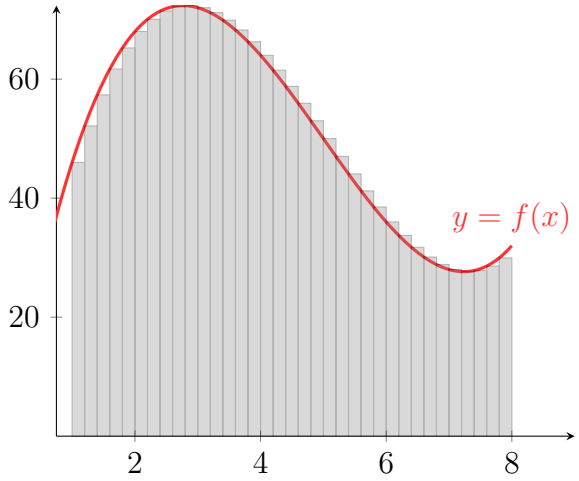


Figura 1.3: Representação da integral numérica calculada através do método da construção de retângulos

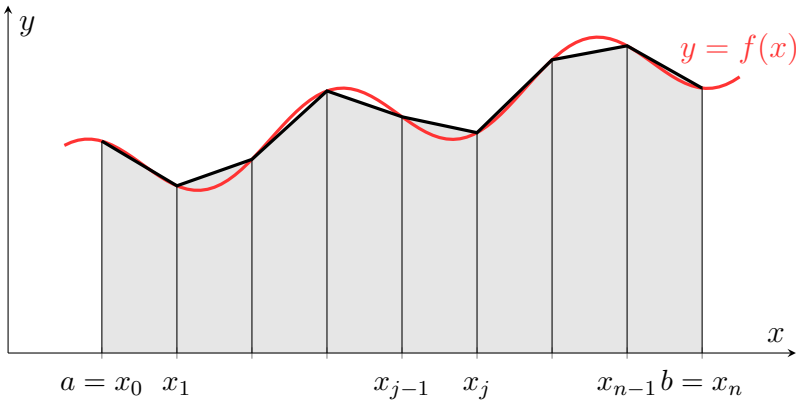


Figura 1.4: Representação da integral numérica calculada através do método de construção de trapézios

é possível notar que a diferença entre as duas equações (1.9) e (1.10) é o termo $-\frac{1}{2}(f_1+f_N)$, este termo corresponde a nossa melhora no valor da integral, porem se houver um grande número N esse termo se torna desprezível.

Durante as aproximações para para calcular a integral numérica tanto com retângulos na equação (1.9) quanto na equação (1.10) fixamos um mesmo valor para Δx_i, porem o valor de Δx_i não precisa ser necessariamente o mesmo.

Como foi definido a integral numérica agora é possível estudar transformada de Fourier.

1.3.2 Transformada de Fourier

A seguir usaremos a letra t para indicar a variável independente da função $f(t)$, porém t não precisa necessariamente representar o tempo, na transformada de Fourier termos como variável independente ω "frequência angular", $\omega = 2\pi\nu$, onde ν é a frequência. Caso a variável t ser espacial o termo ω representa o "número de onda", $k = 2\pi\lambda$, no caso λ é o "comprimento de onda".

Seja $f(t)$ uma função tal que [4]

$$g(\omega) = \int_{-\infty}^{\infty} e^{-(i\omega t)} f(t) d(t) = \mathcal{F}[f(t)], \tag{1.11}$$

neste caso a função $g(\omega)$ representa a transformada de Fourier da função $f(t)$.

Sua transformada inversa pode ser calculada da seguinte forma

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{(i\omega t)} g(\omega) d\omega \ . \tag{1.12}$$

A transformada de Fourier apresenta algumas propriedades [6], tais como:

Paridade

a) Caso $f(t)$ ser real, então

$$\mathrm{Re}[g(-\omega)] = \mathrm{Re}[g(\omega)]$$
$$\mathrm{Im}[g(-\omega)] = -\mathrm{Im}[g(\omega)]$$

ou seja

$$g(-\omega) = g^*(\omega) \tag{1.13}$$

b) Se $f(t)$ é real e par, então $g(\omega)$ é real;

c) Se $f(t)$ é real e ímpar, então $g(\omega)$ é imaginária;

d) Se $f(t)$ é imaginária então as propriedades a, b, c, acima, serão validas se trocarmos real por imaginária e vice e versa.

Transformada da Convolução

E possível definir a convolução da função $f_1(t)$ com a função $f_2(t)$ da seguinte forma

$$(f_1 \otimes f_2) = \int_{-\infty}^{\infty} f_1(t) f_2(\tau - t) d\tau, \tag{1.14}$$

calculando a transformada de Fourier da convolução temos

$$\mathcal{F}[f_1 \otimes f_2] = \mathcal{F}[f_1]\mathcal{F}[f_2] = g_1(\omega) g_2(\omega). \tag{1.15}$$

Transformada da Autocorrelação

A função de autocorrelação para a função funções $f(t)$ é definida da seguinte forma

$$Corr(f(t), f(t)) = \int_{-\infty}^{\infty} f(t) f(\tau + t) d\tau, \tag{1.16}$$

A função de autocorrelação, é uma medida que informa como a função se relaciona com ela mesmo ao longo [7]. O valor da autocorrelação está entre 1 (correlação perfeita) e -1, o que significa anticorrelação perfeita o valor 0 significa total ausência de correlação.

A autocorrelação de uma dada variável se define pela distância, ou atraso com que se deseja medi-la. Quando essa distância é zero, tem-se o valor máximo 1, pois trata-se da variável correlacionada com ela mesma. Outros valores devem ser calculados caso a caso. A função de autocorrelação mede o quanto uma função se relaciona com ela mesmo ao longo do tempo ou do espaço.

Apesar da semelhança com a convolução é possível notar que a variável de integração, t aparece com o sinal positivo em ambos os argumentos da função $f(t)$, diferentemente da convolução onde a variável t aparece com sinal negativo no argumento da segunda $f(t)$ (1.14).

Usando da propriedade da paridade (1.13) temos a transformada de Fourier como

$$\mathcal{F}[Corr(f(t), f(t)] = g(-\omega)g(\omega) = g^*(\omega)g(\omega). \tag{1.17}$$

Para a passagem anterior estamos supondo que a função $f(t)$ é real, note que transformada de Fourier da Correlação não é igual ao produto das transformadas de Fourier da Correlação.

Transformada de Fourier Discreta

Nosso trabalho depende da utilização de redes, neste sentido é importante discretizar a Transformada de Fourier.

A discretização da Transformada de Fourier será feita utilizando a regra do trapézio. Supondo agora que o número de termos n da integral discretizada é suficientemente grande para que possamos desprezar a correção nas pontas $\frac{1}{2}(f_1 + f_n)$ assim

$$g(w_k) = \mathcal{F}[f_j] = \Delta t \sum_{j=1}^{N} e^{(-i\omega_k t_j)} f_j. \tag{1.18}$$

Eis que agora surge uma pergunta "Como definir os valores de ω_k?". Como queremos manter a relação bi-unívoca entre $f \leftrightarrow g$, assim devemos usar o mesmo número n de ω_k que temos em t_j. Para se caracterizar uma variação na função $f(t)$ é preciso realizar um deslocamento mínimo de seu valor original f_j. Esse deslocamento irá afetar a função $f(t)$ num intervalo $\tau = 2\Delta t$. Assim este é o mínimo período de oscilação que pode ser visto em $g(\omega)$. A maior frequência relevante será, $\nu = \frac{1}{2\Delta t}$. Assim temos que $\omega_n = 2\pi\nu = \frac{\pi}{\Delta t}$.

Vamos reescrever a equação (1.18) na forma matricial. Para isto definimos a matriz quadrada (W) de elementos.

$$W_{kj} = e^{(-i\omega_k t_j)},$$

e os vetores coluna f, de elementos f_j e g, de elementos g_k. Com isto a equação (1.18) fica

$$g = W * f. \tag{1.19}$$

Para se calcular a transformada inversa basta fazer o produto com a matriz inversa W^1.

$$f = W^{-1} * g. \tag{1.20}$$

Transformada de Fourier Rápida

Para funções $f(t)$ muito complexas, ricas em detalhes é necessário um número muito grande de N na discretização de t para garantir um gráfico fiel. Este problema se torna maior quando a variável independente tem duas ou três dimensões. Por exemplo, se t é espacial e é representada como $t \to r$.

Existe um algoritmo rápido e eficiente para contornar o problema, hoje ele é conhecido por "FFT" (*Fast Fourier Transform*), no que se segue iremos demonstrar brevemente a

ideia do algoritmo. Vamos tratar agora N como sendo uma potencia de 2, ou seja $N = 2^n$, onde n é um número inteiro. Caso N não seja uma potencia de dois completamos $f(t)$ com zeros até que N seja uma potencia de 2. Para facilitar é tomado $\Delta t = 1$ e as frequências angulares ω entre $[0, 2\pi)$. A seguir será feito uso das seguintes notações $f_{_j} = f(t_{_j})$ e $g_{_k} = g(\omega_{_k})$, com $\omega_{_k} = 0, \frac{2\pi}{N}, ..., \frac{2k\pi}{N}...$ Assim reescrevendo a equação (1.19) temos

$$g_{_k} = \sum_{j=0}^{N-1} [\exp(\frac{-2\pi i}{N})]^{kj} f_{_j} \ , \tag{1.21}$$

Como N é par podemos separar a somatória na parte e impar e par, cada parte contendo $\frac{N}{2}$ termos:

$$g_{_k} = \sum_{j=par}^{N-2} [\exp(\frac{-2\pi i}{N})]^{kj} f_{_j} + \sum_{j=impar}^{N-1} [\exp(\frac{-2\pi i}{N})]^{kj} f_{_j} =$$

$$\sum_{j=0}^{\frac{N}{2}-1} [\exp(\frac{-2\pi i}{N})]^{k(2j)} f_{_{2j}} + \sum_{j=0}^{\frac{N}{2}-1} [\exp(\frac{-2\pi i}{N})]^{k(2j+1)} f_{_{2j+1}} \ . \tag{1.22}$$

Denotando $W = \exp(\frac{-2\pi i}{N})$ e $N' = \frac{N}{2}$, reescrevendo a equação (1.22) temos

$$g_{_k} = \sum_{j=0}^{N'-1} [\exp(\frac{2\pi i}{N'})]^{kj} f_{_{2j}} + W^k \sum_{j=0}^{N'-1} [\exp(\frac{2\pi i}{N'})]^{kj} f_{_{2j+1}}, \tag{1.23}$$

$$g_{_k} = g_{_k}^{\text{par}} + W^k g_{_k}^{\text{ímpar}}, \tag{1.24}$$

onde $g_{_k}^{par}$ é a soma de todos os termos pares e $g_{_k}^{impar}$ é a soma de todos os termos ímpares, neste sentido acima a transformada de Fourier é dividida em 2 partes, um com pontos pares e outro com os pontos ímpares. Assim a equação (1.19) passa a conter $2N'^2 = \frac{N^2}{2}$ multiplicações em vez dos N^2 que tínhamos no início. Como havíamos definido N como sendo uma potência de 2 ($N = 2^n$) temos que $N' = 2^{n-1}$. Podemos separar a transformada de Fourier (1.24) em duas, cada uma para $\frac{N}{4}$. Dando continuidade ao processo até que a transformada se torne um único ponto. Pode se verificar que o processo todo terá $N \log(N)$ multiplicações em vez de N^2 que tínhamos no inicio. Se tivermos um número N muito grande é possível reduzir dias de computação para segundos [4].

Apesar de todos os benefícios a "fft" (Fast Fourier Transform) apresenta alguns inconvenientes que podem ser contornados. As frequências da fft são definidas entre $[0, 2\pi)$ com isso as componentes de baixa frequência da transformadas são encontradas no centro do gráfico e não nas extremidades. Além disso, por usar $\delta t = 1$ a transformada não informa sobre os valores das frequências, registrando na abscissa do gráfico apenas os índices das componentes de $\omega_{_k}$ do vetor das frequências. Todos esses problemas podem ser corrigidos usando as propriedades da transformada de Fourier.

1.4 Método de Euler e Runge-Kutta.

No que se segue ira ser apresentado o método com o qual as equações diferenciais serão resolvidas durante as simulações.

1.4.1 Método de Euler

Seja t uma variável real definida entre $[0, t_{max}]$. Iremos obter uma solução numérica para uma equação diferencial ordinária de primeira ordem [4].

$$\frac{dx}{dt} = \dot{x} = f(x,t) \ . \tag{1.25}$$

O primeiro passo a ser tomado no procedimento numérico é discretizar a variável contínua t e substitui-la

$$t_1 = 0, \quad t_2 = t_1 + \Delta t, \quad t_3 = t_2 + \Delta t, ..., t_n = t_{max} \ ,$$

em alguns casos o intervalo Δt pode não ter um valor fixo.

Fazendo uso da seguinte notação $\dot{x}$ para indicar derivada em relação a variável t, a derivada numérica é definida pela equação (1.3) sem o termo de erro

$$\dot{x}_j = \frac{x_{j+1} - x_j}{\Delta t}, \tag{1.26}$$

e a forma mais precisa usando a equação (1.5)

$$\dot{x}_j = \frac{x_{j+1} - x_{j-1}}{2\Delta t}, \tag{1.27}$$

reescrevendo a equação (1.26) junto com a equação (1.25) temos

$$x_{j+1} = x_j + \Delta t f(x_j, t_j). \tag{1.28}$$

O método apresentado anteriormente para a resolução numérica de equações diferenciais de primeira ordem é conhecido como **Método de Euler**.

Analogamente é possível escrever a equação (1.27) em conjunto com com a equação (1.25)

$$x_{j+1} = x_{j-1} + 2\Delta t f(x_j, t_j). \tag{1.29}$$

1.4.2 Método de Runge-Kutta

A fim de melhorar a precisão da solução numérica calculada anteriormente no que se segue será apresentado o método de Runge-Kutta.

Ao retornar para a equação (1.28)

$$x_{j+1} = x_j + \Delta t f(x_j, t_j), \tag{1.30}$$

ao fazer uso da simetria, é possível reescrever a equação (1.28) para o argumento da f ao extremo á direita do intervalo $[t_j, t_{j+1}]$

$$x_{j+1} = x_1 + \Delta t f(x_{j+1}, t_{j+1}), \tag{1.31}$$

para obter uma solução melhor que a equação (1.5) e a equação (1.31) será tomado o valor intermediário da função f,

$$x_{j+1} = x_j + \Delta t f(\frac{x_j + x_{j+1}}{2}, \frac{t_j + t_{j+1}}{2}). \tag{1.32}$$

Embora algum progresso tenha sido feito surge um novo problema, o valor de x_{j+1} dentro do argumento da f é desconhecido. Para contornar este problema será feito o uso da a equação (1.31) para obter uma primeira aproximação de x_{j+1} a se usar no argumento da f

$$x_{j+1} = x_j + \Delta t f(x_j + 0,5\Delta t f(x_j, t_j), t_j + 0,5\Delta t), \tag{1.33}$$

a equação (1.33) é chamada de **Runge-Kutta de segunda ordem**, ela também é conhecida como **Euler modificada**.

1.4.3 Runge-Kutta de Quarta Ordem

É possível melhorar ainda mais a precisão, no que se segue será apresentado o método de Runge-Kutta de quarta ordem [4].

Agora iremos abordar o método de **Runge-Kutta de quarta ordem**, o método em si se baseia em usarmos mais pontos intermediários no intervalo de $[t_j, t_{j+1}]$. Vamos agora considerar a equação (1.25) definindo agora $h = \frac{\Delta t}{2}$, o esquema é

$$\begin{aligned} F_1 &= f(x_j, t_j) \\ F_2 &= f(x_j + hF_1, t_j + h) \\ F_3 &= f(x_j + hF_2, t_j + h) \\ F_4 &= f(x_j + hF_3, t_j + \Delta t) \\ x_{j+1} &= x_j + \frac{\Delta t}{6}(F_1 + F_2 + F_3 + F_4) \end{aligned} \tag{1.34}$$

A principio é possível construir métodos de **Runge-Kutta** de qualquer ordem, para isso basta adicionar mais passos ao método, porém quanto mais passos o método possui maior é o número de vezes que se necessita calcular $f(x, t)$, assim aumentando, ainda mais o tempo gasto. Para a maioria dos casos o método de segunda ordem já é o suficiente, além de possuir uma precisão muito boa ele é relativamente rápido de se calcular.

Capítulo 2

R.P.S.

Neste capítulo será abordado as regras do jogo RPS (Pedra-Tesoura-Papel), que serão usadas em nossas simulações.

2.1 Pedra, Papel, Tesoura

Para começar vamos imaginar três espécies (A, B, C) interagindo entre si de maneira cíclica com as seguintes regras a espécie A vence da espécie B, a espécie B vence a espécie C, e a espécie C vence a espécie A. O jogo RPS ilustra bem está competição cíclica.

No jogo RPS existe uma interação cíclica bem definida, a pedra sempre vence da tesoura que por sua vez sempre vence do papel e o papel sempre vence da pedra conforme ilustra a figura 2.1. Existem variações do jogo pedra, papel e tesoura uma variação muito conhecida é o pedra, tesoura, papel, lagarto, Spock conforme ilustra a figura 2.2, apesar de ter adicionado mais duas espécies a competição as mesmas obedecem a uma interação cíclica. Também existem estudos generalizado para n espécies interagindo [8]. Contudo o foco neste trabalho é o modelo com três espécies.

Figura 2.1: Ilustração da interação cíclica do jogo pedra tesoura e papel

Dando continuidade ao estudo com três espécies, cada espécie irá representar uma população de indivíduos, estas espécies estarão distribuídas de maneira aleatória em uma rede bidimensional de N colunas e linhas. Esta rede possui com condições de contorno periódicas, com N^2 sítios distribuídos na mesma conforme ilustra a figura. 2.3.

No que se segue será apresentado dois modelos para a Dinâmica de Populações, o Modelo Estocástico e o Modelo de May-Leonard. Porém existem na literatura estudos com modelos diferentes [9].

Antes de abordarmos os dois modelos, precisamos estabelecer o conceito de vizinhança. Iremos usar a vizinhança de Neumann neste trabalho. A vizinhança de Neumann consiste em quatro células ortogonais ao redor de uma celular bidimensional [10].

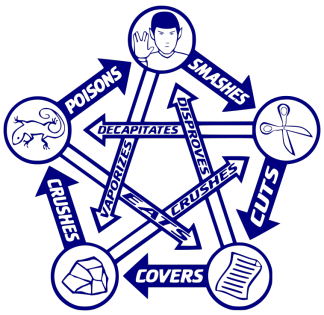


Figura 2.2: Ilustração da interação cíclica no jogo pedra, tesoura, papel, lagarto e Spock

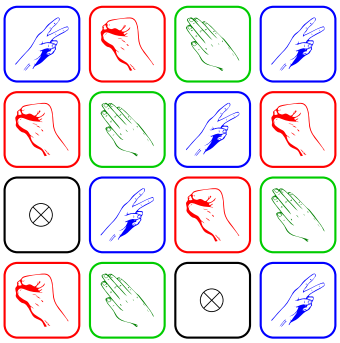


Figura 2.3: Rede bidimensional com 4 linhas e colunas

2.2 Modelo Estocástico

No jogo RPS estocástico unidirecional cada espécie é representada por um estado (A, B, C) e por sítios vazios ($\otimes$), onde cada elemento pode predar e ser predado com uma probabilidade p. Existem também variações do modelo estocástico focando na interação simétrica presa e predador [11], mas o foco deste trabalho é o modelo unidirecional.

Cada passo na rede é regido pela seguinte diretriz.

1. Um elemento da rede é sorteado, esse elemento recebe o nome de ativo

2. Um dos vizinhos mais próximos do elemento ativo é sorteado, o elemento recebe o nome de passivo.

3. Uma interação é sorteada (predação, reprodução, mobilidade) obedecendo os respectivos pesos definidos inicialmente

4. Caso a interação sorteada seja predação é verificado se o passivo pode ser predado, se sim o elemento passivo se torna um sítio vazio, se não nada acontece, a Figura 2.4 demonstra uma interação do tipo predação

5. Caso a interação de mobilidade seja sorteada ocorre a troca de posição entre o ativo e passivo, a Figura 2.5 demonstra uma interação do tipo mobilidade

6. Caso a interação seja reprodução é verificado de se o elemento passivo é um sitio vazio se sim o individuo é duplicado, se não nada acontece a figura 2.6 demonstra uma interação do tipo reprodução.

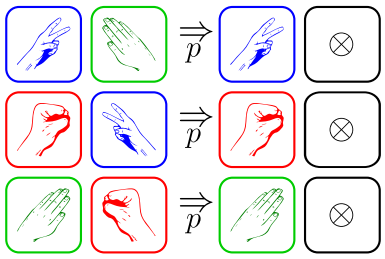


Figura 2.4: Demonstração de uma interação do tipo predação.

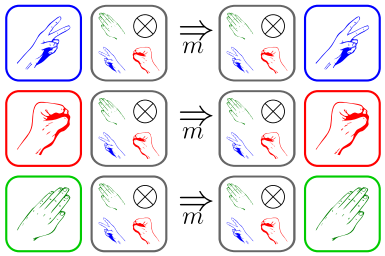


Figura 2.5: Demonstração de uma interação do tipo mobilidade.

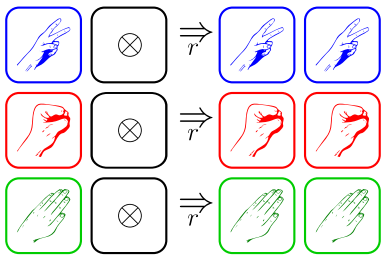


Figura 2.6: Demonstração de uma interação do tipo reprodução.

A Figura 2.7 demonstra a evolução de uma rede após sete passos temporais. Foram sorteados sete elementos ativos, passivos, interações, e verificações de possibilidade de execução destas interações sorteadas.

Neste modelo a Geração é definida como sendo o número de passos temporais igual a N^2, em média cada indivíduos da rede é sorteado uma vez a cada geração.

2.3 Modelo de Campo Médio

Na secção anterior foi apresentado o modelo estocástico. Neste modelo todas as ações e escolhas eram por processos aleatórios. O modelo que irá ser estudado a seguir é o modelo no qual todas as interações entre as espécies serão feitas por equações diferenciais. O modelo a ser estudado a seguir é conhecido como modelo de May-Leonard, este modelo incorpora as principais regras associadas a dinâmica de competição entre N espécies. No modelo simplificado, três espécies interagem entre si, cada espécie pode realizar três ações

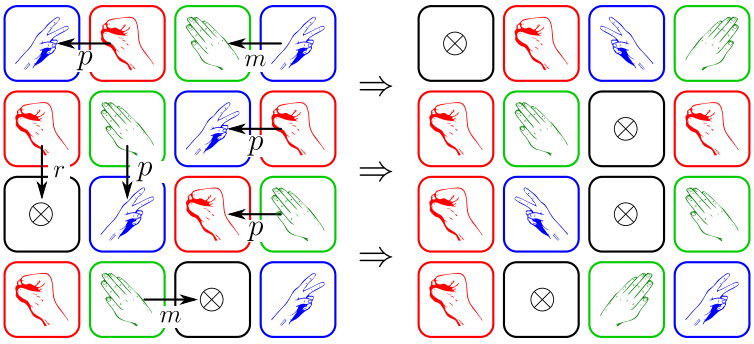


Figura 2.7: A esquerda a rede dada como condição inicial, a direita temos a mesma rede após sete passos temporais.

(mobilidade, reprodução e predação). No entanto o modelo de May-Leonard não se limita a somente uma competição entre três espécies, existem generalizações na literatura com n espécies diferentes [11].

Neste trabalho será estudado o modelo com três espécies. Cada espécie será representada por um campo escalar (ϕ_1, ϕ_2, ϕ_3) e um campo (ϕ_0) que representa o vazio, distribuídos de maneira aleatória em uma rede, a evolução temporal de cada um dos campos é representada da seguinte forma,

$$\dot{\phi_i} = D_i \nabla^2 \phi_i + r_{i,0} \phi_i \phi_0 - \sum_{i=1}^{3} p_{i,i+1} \phi_i \phi_{i+1} \tag{2.1}$$

para as três espécies e

$$\dot{\phi_0} = D_0 \nabla^2 \phi_0 - \sum_{i=1}^{3} r_{i,0} \phi_i \phi_0 + \sum_{i=1}^{3} p_{i,i+1} \phi_i \phi_{i+1} \tag{2.2}$$

para representar o vazio.

O termo D_i é a constante difusiva da espécie, $r_{i,0}$ é constante reprodutiva da espécie i , e $p_{i,i+1}$ é a constante de predação da espécie i com a espécie $i+1$.

O termo que contém o Laplaciano $(D\nabla_i^2 \phi_i)$ representa a ação de mobilidade da espécie dentro da rede. O termo $r_{i,0}\phi_0\phi_i$ representa a ação de reprodução dentro da rede. Note que para que qualquer espécie se reproduza é preciso que haja vazio no sítio. Note também que aqui é diferente do modelo estocástico, o vazio é tratado como uma espécie enquanto no modelo estocástico não. O papel da espécie vazio é evitar que haja na rede superpopulações em sítios. Por fim o termo $p_{i,i+1}\phi_i\phi_{i-1}$ representa o termo de predação de uma espécie com a outra. Neste termo temos uma interação entre duas espécies diferentes.

Em nosso modelo $r_{i,0} = r$ para as todas as espécies, e $p_{i,i+1} = p$ para as todas espécies, adotando o modelo de predação unidirecional assim como no jogo pedra papel e tesoura. Existem na literatura modelos com sentido bidirecional [11], mas não é o foco neste trabalho. Reescrevendo a equação (2.1) para todas as espécies temos

$$\dot{\phi_1} = D\nabla^2 \phi_1 + r\phi_1\phi_0 - p\phi_1\phi_2 \tag{2.3}$$

$$\dot{\phi_2} = D\nabla^2 \phi_2 + r\phi_2\phi_0 - p\phi_2\phi_3 \tag{2.4}$$

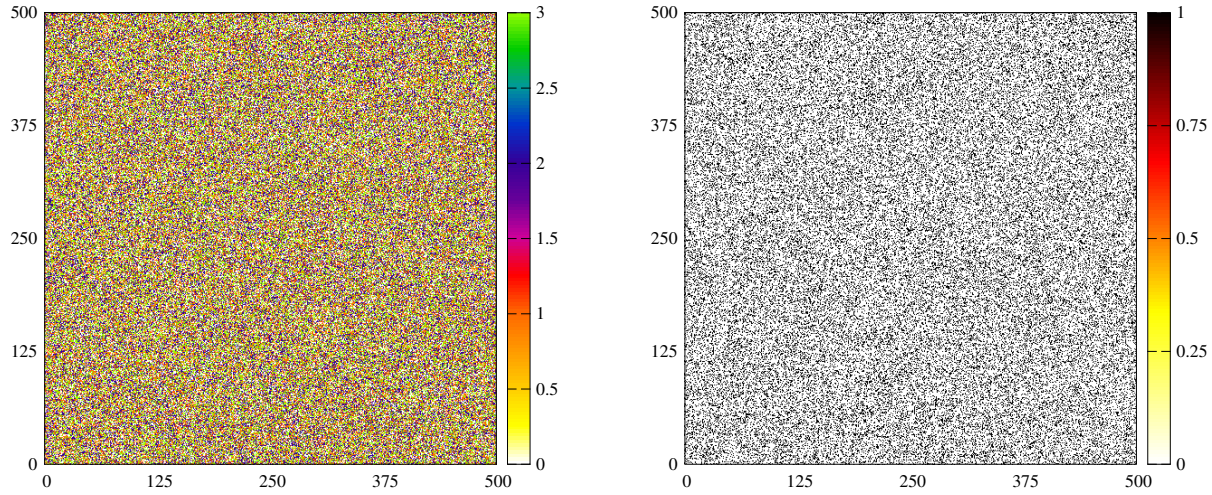
$$\dot{\phi_3} = D\nabla^2 \phi_3 + r\phi_3\phi_0 - p\phi_3\phi_1 \tag{2.5}$$

$$\dot{\phi_0} = D\nabla^2 \phi_1 - r\phi_1\phi_0 - r\phi_2\phi_0 - r\phi_3\phi_0 + p\phi_1\phi_2 + p\phi_2\phi_3 + p\phi_3\phi_1 \ . \tag{2.6}$$

Capítulo 3

Resultados

No decorrer deste capítulo serão expostos os resultados das simulações feitas com o Modelo Estocástico e com o Modelo de May-leonard. Em todas as simulações foi usado uma rede de 500 × 500, cada espécie foi distribuída na rede de maneira aleatória conforme ilustra a figura 3.1, cada simulação foi evoluída por com 10 mil gerações.



(a) $t = 0$ (b) $t = 0$

Figura 3.1: A esquerda a rede gerada aleatoriamente, a direita a mesma rede mas somente para os espaços vazios

3.1 Modelo Estocástico

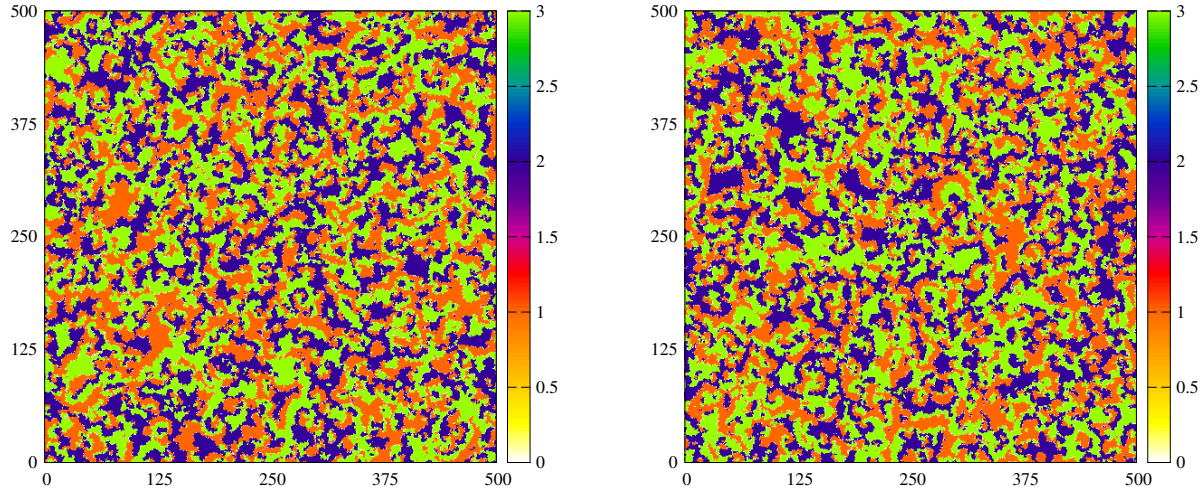
No que se segue serão apresentadas simulações com o Modelo Estocástico. Em nossas simulações iremos usara seguinte equação para definir os valores de r , p e m

$$r + p + m = 1,$$

onde p, r, e m são respectivamente predação, reprodução e mobilidade.

Iremos começar estudando a ação da mobilidade na rede. Para a primeira simulação será adotado $m = 0$, $p = 0,5$ e $r = 0,5$.

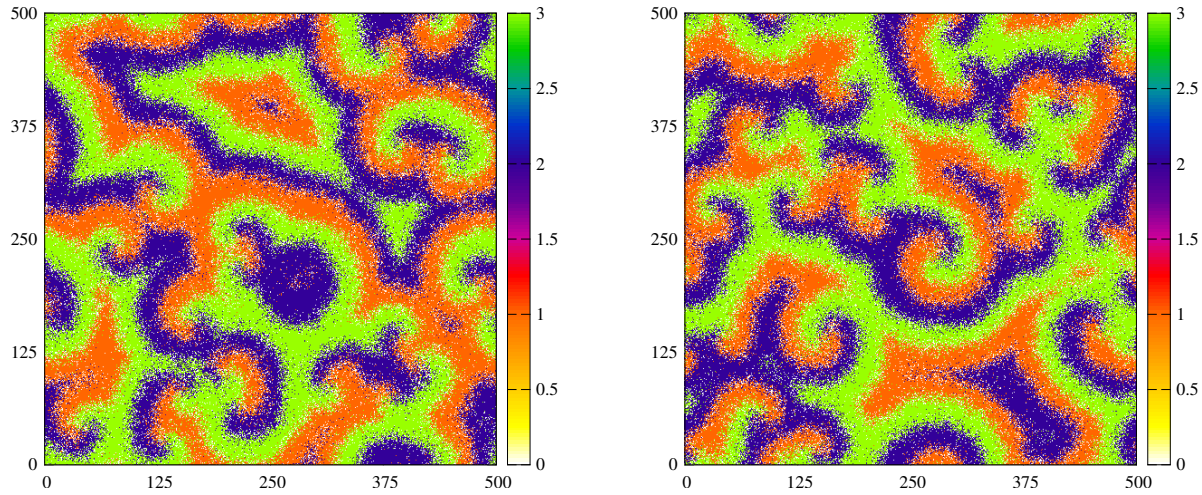
A primeira conclusão é que a rede evolui, a rede não forma nenhum padrão, porém ela forma algumas ilhas de espécies conforme mostra a figura 3.2.



(a) $t = 5000$ (b) $t = 10000$

Figura 3.2: Nesta sequência temporal de imagens é mostrado a evolução da rede estocastica com $m = 0$, $p = 0,5$ e $r = 0,5$.

Para a segunda simulação será introduzido a ação de mobilidade na rede. Usando os seguintes parâmetros $m = 0,8$, $p = 0,1$, e $r = 0,1$.



(a) $t = 5000$ (b) $t = 10000$

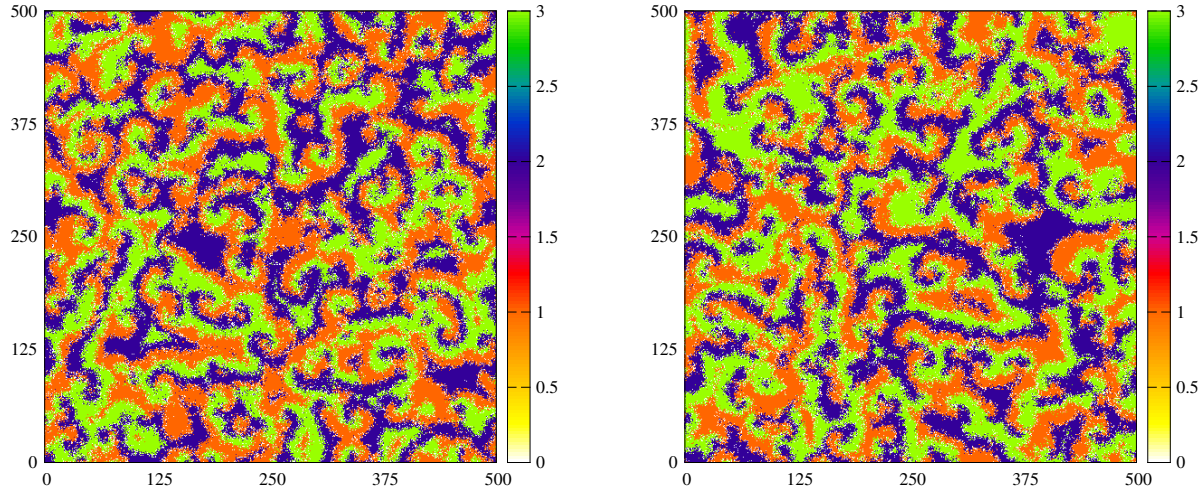
Figura 3.3: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $m = 0,8$, $p = 0,1$ e $r = 0,1$.

É possível identifica que na figura 3.3b há a formação de padrões espirais nas simulações.

Para estudar o quanto a ação de mobilidade influencia nas espirais será feitos mais uma simulação. Na próxima simulação foi usado $m = 1/3$, $p = 1/3$ e $r = 1/3$.

E notável que na figura 3.4 o tamanho das espirais também diminui. Assim é possível notar que m está ligado ao tamanho das espirais. Ao aumentar m aumentamos as espirais. Diminuir m diminui as espirais.

No que se segue será estudado como os parâmetros de reprodução r e predação p

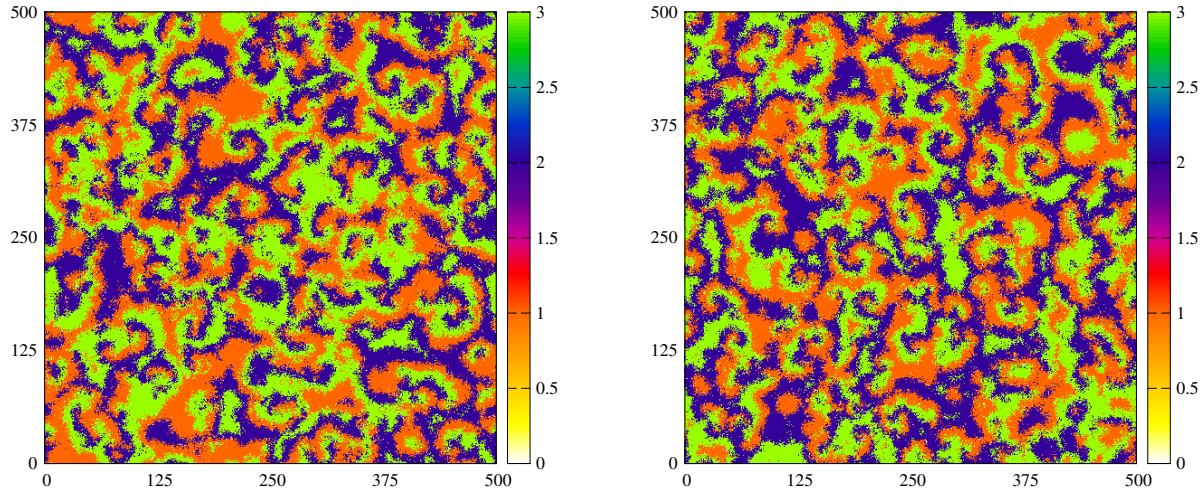


(a) $t = 5000$ (b) $t = 10000$

Figura 3.4: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $m = 1/3$, $p = 1/3$ e $r = 1/3$.

interferem na rede.

Para a quarta simulação foi tomado $m = 0,1$, $p = 0,1$ e $r = 0,8$.

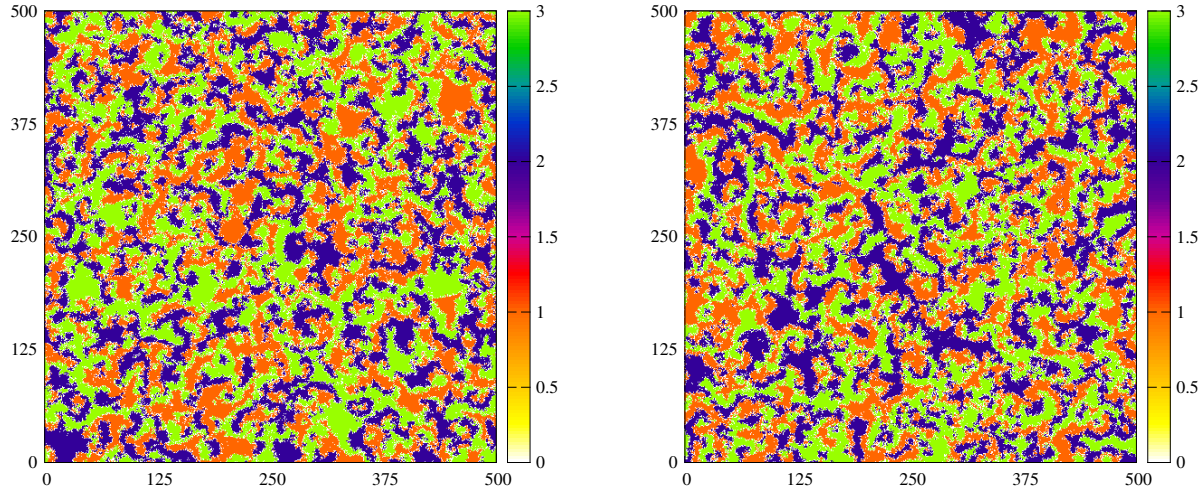


(a) $t = 5000$ (b) $t = 10000$

Figura 3.5: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $m = 0,1$, $p = 0,1$ e $r = 0,8$.

Para comparar com a simulação anterior foi realizado uma nova simulação, mas agora com os valores de r e p alternados.

E notável que ao comparar a Figura 3.5b com a Figura 3.6b é que na ultima não há a formação de padrões de espirais bem definidos. Na quarta simulação os padrões de espirais formam-se de maneira bem definida. A instabilidade gerada na quinta simulação se deve ao fato da combinação de uma baixa taxa de mobilidade combinada com uma alta taxa de predação.



(a) $t = 5000$ (b) $t = 10000$

Figura 3.6: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $m = 0,1$, $p = 0,1$ e $r = 0,8$.

3.2 Modelo de May-Leonard

No que se segue serão apresentadas simulações realizadas com o Modelo de May-Leonard. Usaremos a seguinte equação para definir os parâmetros r, p e D

$$\frac{D}{2} + r + p = 1, \tag{3.1}$$

onde r é a constante de reprodução, p é a constante de predação e D é a constante de difusão.

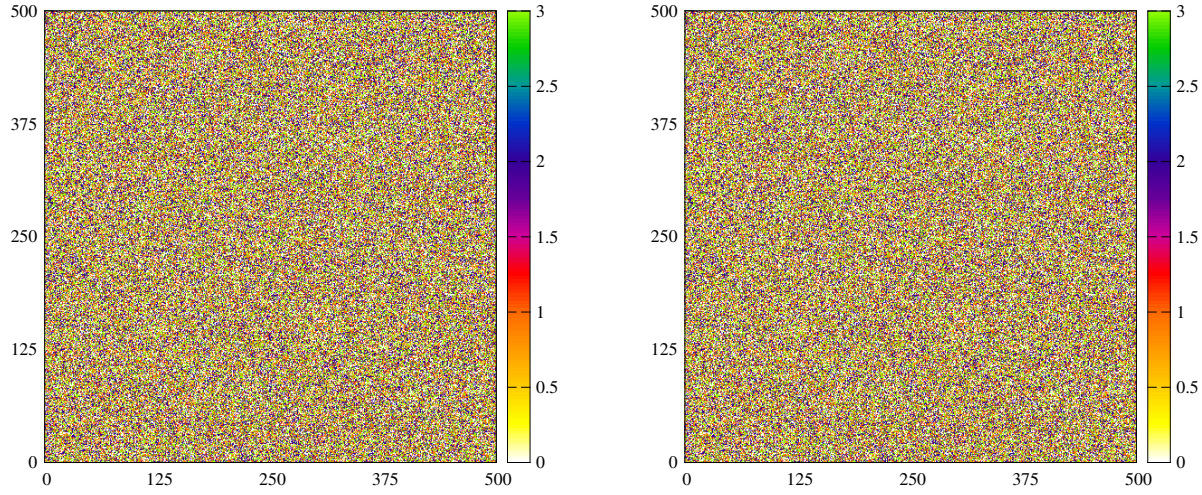
Iremos seguir a mesma ordem de estudos do modelo estocástico. Primeiro iremos estudar a influencia de mobilidade na rede. Posteriormente será estudado como a reprodução e predação interferem na rede.

Para a primeira simulação com o modelo de May-Leonard iremos adotar $p = 0,5$, $r = 0,5$ e $D = 0$,

Pelo que pode ser visto no gráfico 3.7 a rede não evoluiu no tempo. A Figura 3.7a e a Figura 3.7b são iguais, mesmo após 10 mil gerações. Diferente do modelo estocástico, aqui a rede congelou no tempo.

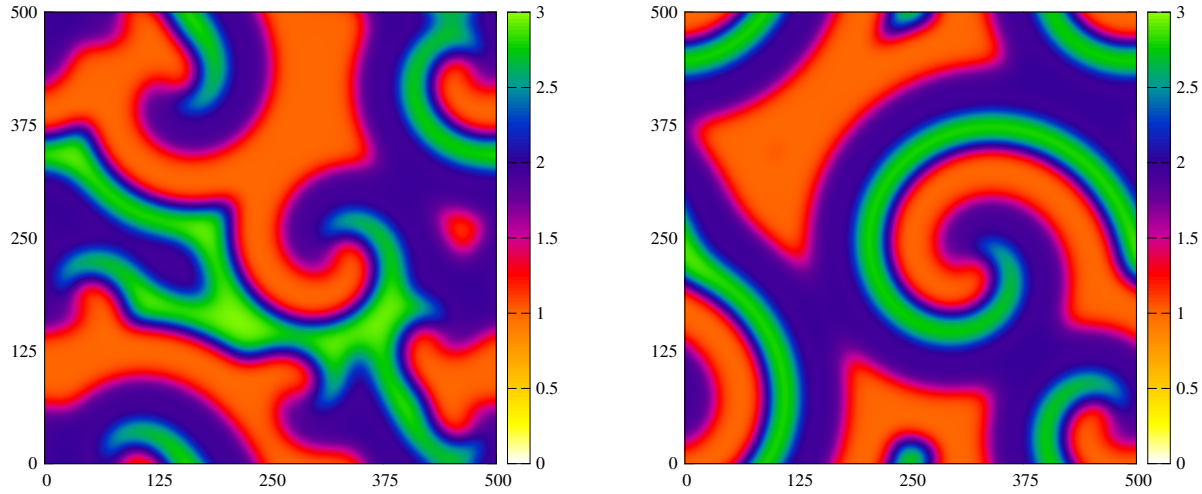
Eis que surge a seguinte pergunta, "Por que a rede não evolui se colocarmos o parâmetro $D = 0$?". Para entender esta situação iremos olhar novamente para as equações (2.3), (2.4), (2.5). Cada equação possui um termo de predação p, reprodução r, e um termo de mobilidade D, para que os termos reprodução e predação consigam agir é preciso que mais de uma espécie coexista em um mesmo sítio. Como o nosso sorteio da rede somente sorteia uma única espécie por sítio, no início da rede somente temos uma espécie por sitio. O único termo da equação que não precisa que duas espécies coexistam em um mesmo sítio é o de mobilidade. Assim se $D = 0$ não existe mobilidade logo não existem duas espécies coexistindo em um mesmo sítio consequentemente a rede não consegue evoluir no tempo.

Para confirmar o que foi dito anteriormente foi realizado uma segunda simulação mas agora com o valor de $D = 1,6$, $p = 0,1$ e $r = 0,1$.



(a) $t = 5000$ (b) $t = 10000$

Figura 3.7: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $D = 0$, $p = 0,5$ e $r = 0,5$.



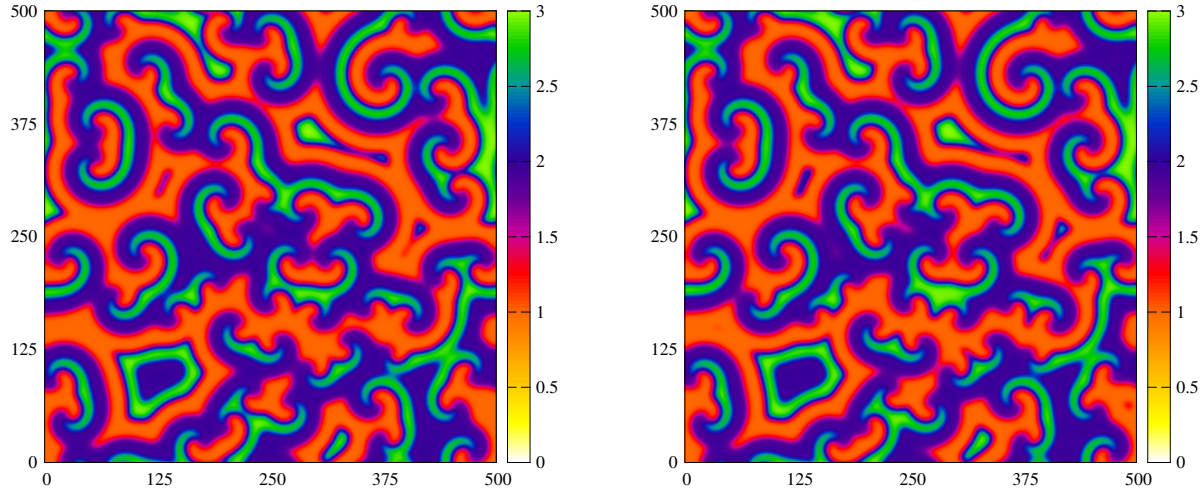
(a) $t = 5000$ (b) $t = 10000$

Figura 3.8: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $D = 1,6$, $p = 0,1$ e $r = 0,1$.

Como é possível analisar na Figura 3.8, com um valor de mobilidade D diferente de 0 a rede evolui no tempo. Além de evoluir no tempo, a rede forma um padrão de espirais com junções triplas. Estas espirais são muito semelhantes as da segunda simulação do modelo estocástico.

A fim de estudar a relação das espirais com o parâmetro difusivo D foi realizado uma nova simulação. Para a simulação novamente foram alterados os valores das constantes para $D = 2/3$, $r = 1/3$ e $p = 1/3$.

É possível notar que na Figura 3.9 a rede continua formando espirais semelhantes as da Figura 3.8, mas agora é notável que o tamanho das espirais mudou. As espirais da figura 3.9a são menores que as da figura 3.8a. Este fato acontece de maneira semelhante no modelo Estocástico. Ao diminuir a mobilidade em ambos os modelos, temos uma



(a) $t = 5000$ (b) $t = 10000$

Figura 3.9: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $D = 2/3$, $p = 1/3$ e $r = 1/3$.

diminuição no tamanho das espirais.

Para a compreensão da mudança do tamanho das espirais vamos analisar o termo relacionado a mobilidade da equação (2.1) usando a derivada segunda na forma numérica

$$\frac{D(\phi_{+h} + \phi_{-h} - 4\phi)}{h^2} \ , \tag{3.2}$$

onde h os termos subscritos $-h$ e $+h$ representam os vizinhos de ϕ.

O termo difusivo D é dividido pelo tamanho do passo h^2. Ao aumentar ou diminuir o valor de D o valor da divisão D/h^2 muda. Ao analisar que ao aumentar o valor de D o valor de h^2 é diminuído a interação de ϕ é realizada com os vizinhos mais próximos. Dessa forma tem se impressão que as espirais formadas são maiores, mas na realidade estamos dando um *zoom* na rede. Analisando o cenário contrario, diminuindo o valor de D, é aumentar o valor de h^2. Aumentando o valor de h^2 faz com que a função ϕ se relacione com pontos mais afastados dando impressão que as espiras são menores, ao fazer isto na realidade estamos nos afastando da rede.

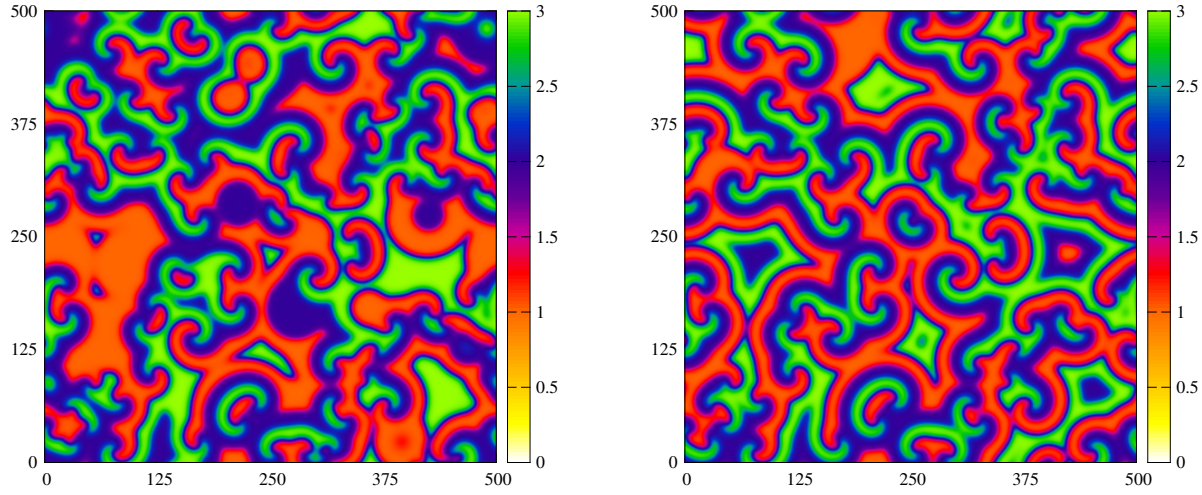
No que se segue serão realizadas simulações para o estudo dos parâmetros de reprodução r e predação p.

Para a quarta simulação iremos usar $D = 0,2$, $p = 0,1$, $r = 0,8$, ou seja o parâmetro $p < r$.

Para comparar com a simulação anterior foi realizado uma quinta simulação mas agora com os valores de r e p alternados $r < p$.

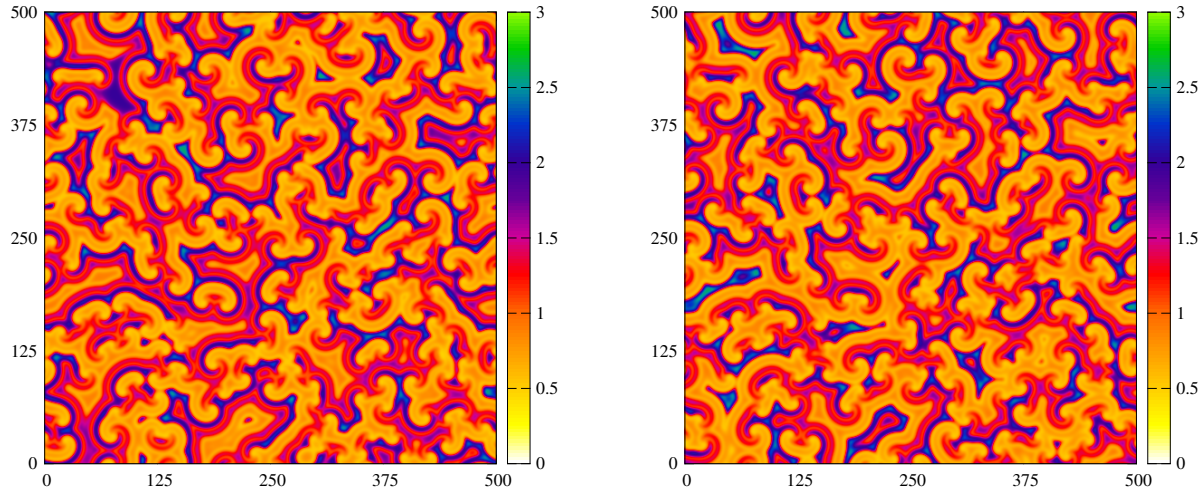
Ao comparar a Figura 3.10 com a Figura 3.11, é notável a diferença de densidade de vazios na rede. Para analisar esta diferença foi confeccionado uma figura apenas com a espécie vazios.

Pelo que se pode notar na Figura 3.12 na Figura 3.13 quando aumentamos a taxa de reprodução em relação a de predação a densidade de vazios na rede diminui. Faz todo sentido afinal com uma taxa de reprodução menor que a taxa de predação a cada geração a densidade de vazios diminui. Em contrapartida ao aumentar a taxa de predação em relação a taxa de reprodução há um aumento na densidade de vazios na rede. É coerente



(a) $t = 2000$ (b) $t = 3000$

Figura 3.10: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $D = 0,2$, $p = 0,1$ e $r = 0,8$.



(a) $t = 5000$ (b) $t = 10000$

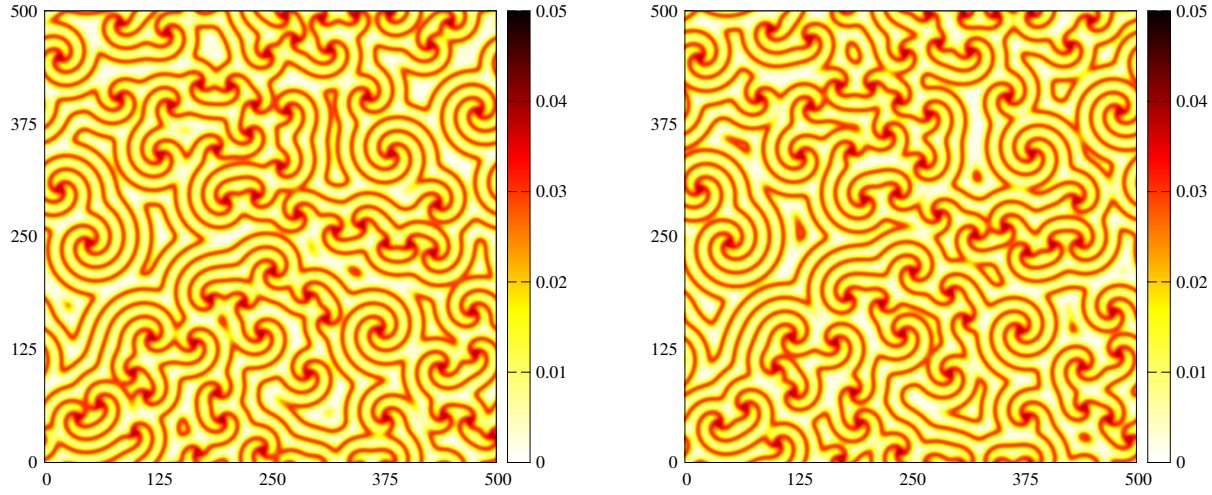
Figura 3.11: Nesta sequência temporal de imagens é mostrado a evolução da rede com os seguintes parâmetro $D = 0,2$, $p = 0,8$ e $r = 0,1$.

1

com a realidade pois aumentando a taxa de predação é criado uma maior densidade maior de vazios, como a taxa de reprodução é pequena há o predomínio da densidade de vazios na rede, vale ressaltar que a densidade de vazios se encontra na interface entre duas espécies diferentes.

3.3 Função de Autocorrelação

A seguir será estudado a função de autocorrelação nas simulações com o Modelo de May- Leonard.



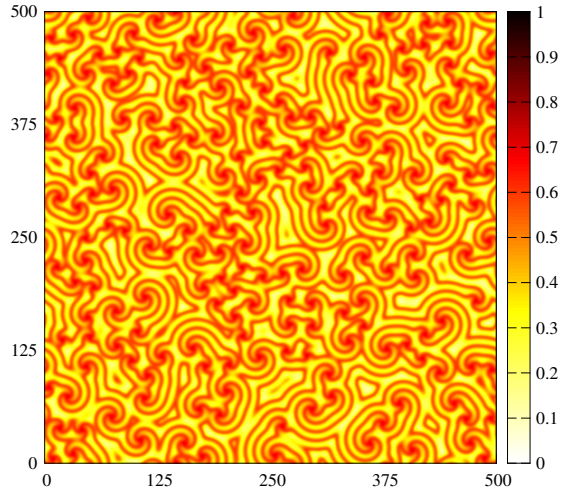
(a) $t = 5000$ (b) $t = 10000$

Figura 3.12: Nesta sequência temporal de imagens é mostrado a evolução da rede com apenas a espécie vazio $D = 0,2$, $p = 0,1$, $r = 0,8$.

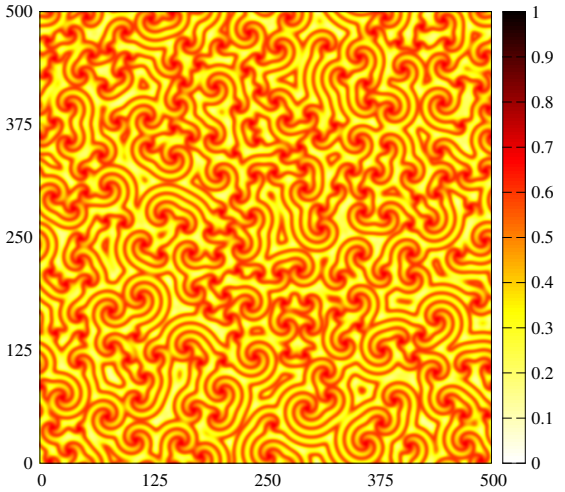
Foram calculada as funções de autocorrelação em todas as simulações com o Modelo de May-Leonard com exceção a primeira simulação, a que foi feita com $D = 0$, pois a rede congelou e não evoluiu.

E perceptível que em todas as Figuras 3.14 a função de autocorrelação apresenta um comportamento semelhante. A medida que a o valor de r aumenta o valor de $|C(r)|$ diminui. Em alguns casos está diminuição acontece de fora mais suave como mostra a figura 3.14a e em outras de maneira mais brusca conforme mostra a figura 3.14d. A seguir foi feito uma confeccionado uma figura com o tamanho das espirais e o valor de $|C(r)|$.

É possível notar a largura das espirais se relacionado com o valor de $|C(r)|$, quanto maior é o valor de $|C|$ maior é largura das espirais. Para analisar o fato apresentado anteriormente, é necessário entender a definição de função de Correlação. A função de autocorrelação mostra como a uma função f se relaciona com ela mesmo ao longo da rede. Uma espiral muito grande, se relaciona com ela mesmo ao longo da rede por uma grande extensão da rede. Para uma espiral muito pequena, a relação come ela mesma ocorre por uma extensão menor durante a rede. Ao calcular a função de correlação nas simulações é perceptível que o valor calculado em $|C(r = 0,5)|$ gera o valor médio da largura das espirais da rede conforme mostra o gráfico 3.15.

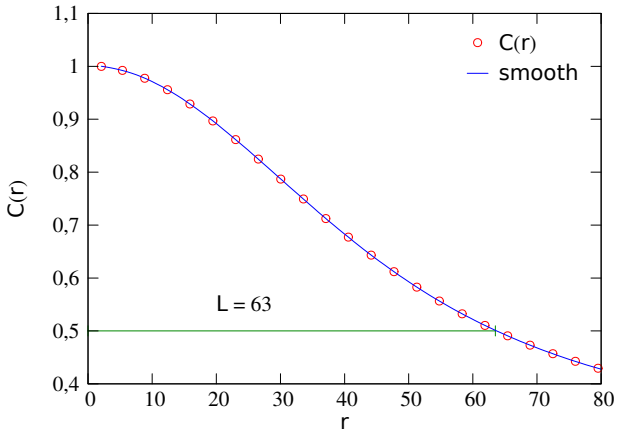


(a) $t = 5000$

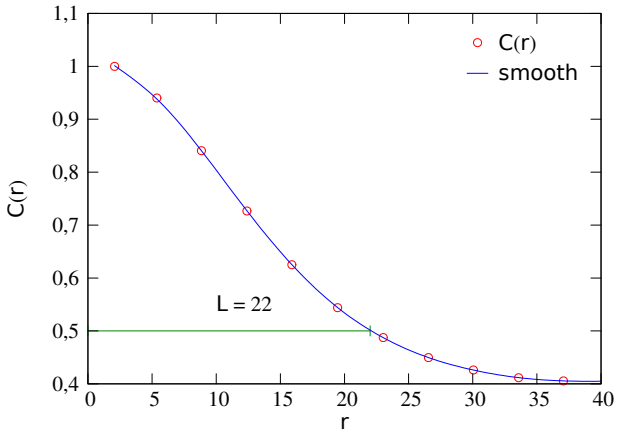


(b) $t = 10000$

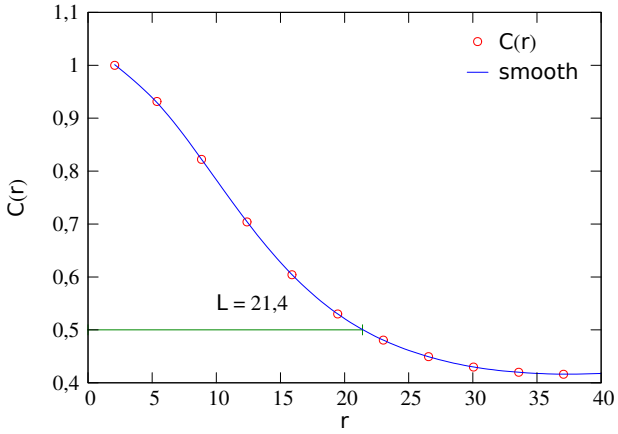
Figura 3.13: Nesta sequência temporal de imagens é mostrado a evolução da rede com apenas a espécie vazio $D = 0,2$, $p = 0,8$ e $r = 0,1$.



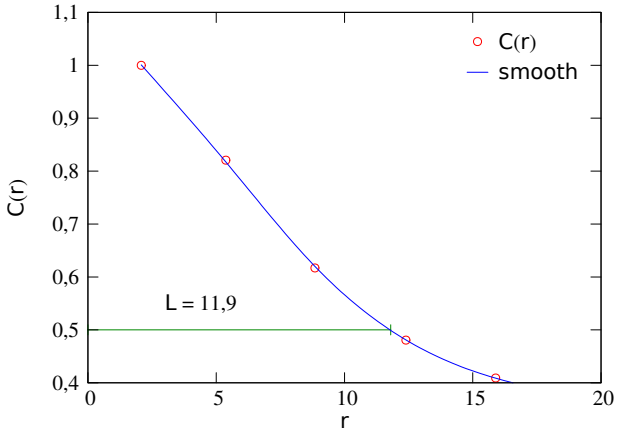
(a) Simulação 2



(b) Simulação 3

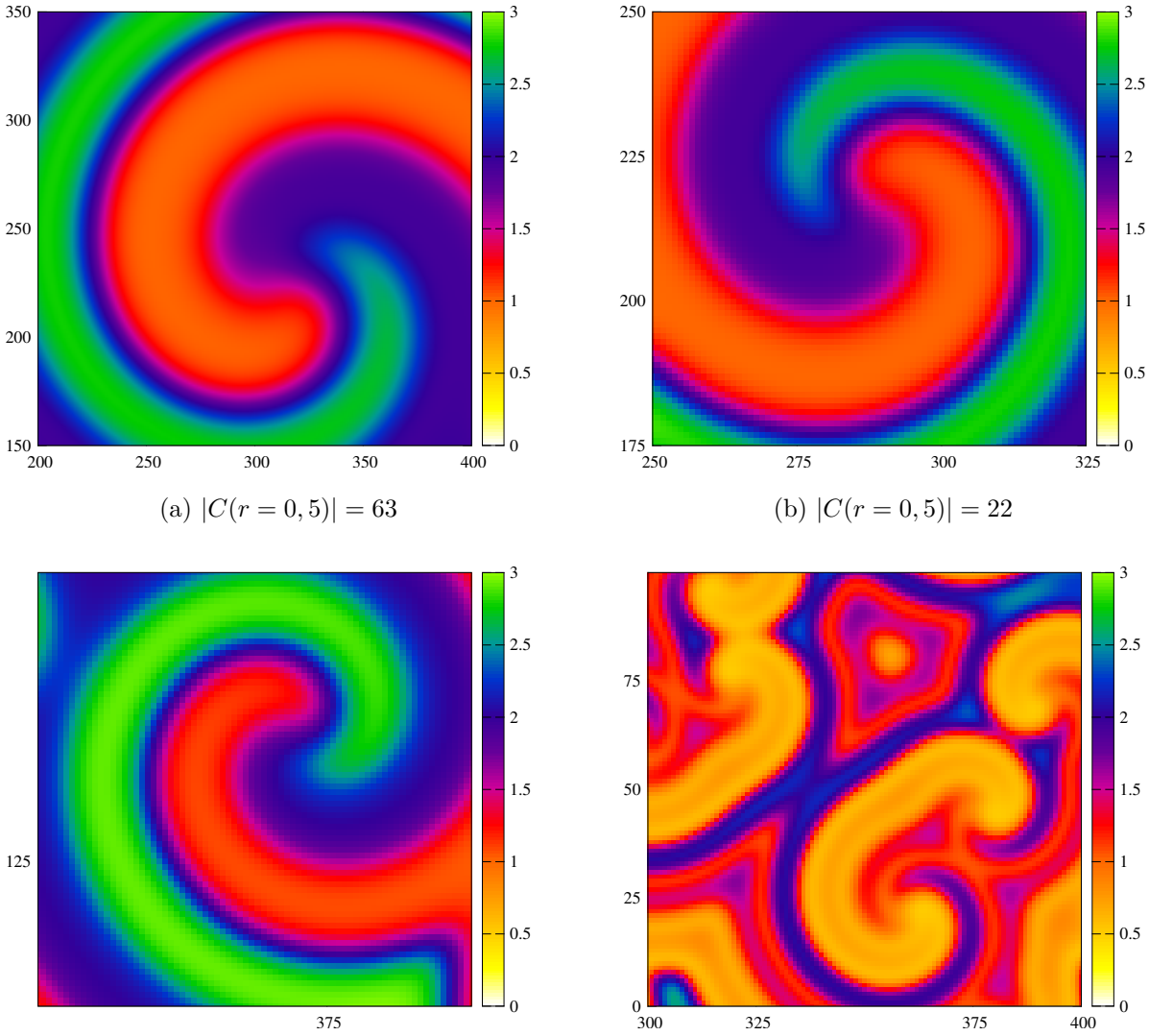


(c) Simulação 4



(d) Simulação 5

Figura 3.14: Nesta sequência imagens é temos os gráficos com a função de correlação em todas as simulações.



(a) $|C(r = 0,5)| = 63$

(b) $|C(r = 0,5)| = 22$

(c) $|C(r = 0,5)| = 21,4$

(d) $|C(r = 0,5)| = 11,9$

Figura 3.15: Nesta sequência imagens é comparado o valor da função de correlação com o tamanho das espirais onde $|C(r)|$ é o valor da função de auto correlação para 0,5.

Conclusões

Neste trabalho estudamos a dinâmica de população por meio do modelo de May-Leonard e do modelo estocástico, ambos seguindo as regras do jogo pedra tesoura e papel. Realizamos simulações com diferentes valores para a mobilidade, partindo da mobilidade zero até uma mobilidade alta. Durante estas simulações foi verificado que o termo de mobilidade gera espirais triplas com três espécies. Em casos em que a mobilidade foi zero vimos que o modelo de May-Leonard e o modelo estocástico divergem, enquanto que no modelo de May-Leonard a rede não evolui no modelo estocástico a rede evolui formando ilhas isoladas de cada espécie. Foi também investigado a relação do tamanho das espirais com o valor da mobilidade, foi visto que quando aumentamos a mobilidade aparentemente aumentamos o tamanho das espirais mas, na realidade aumentando a mobilidade estamos dando uma ampliação na rede. O mesmo vale para o caso contrario, quando diminuímos a mobilidade. Além da ação de mobilidade também foi estudado o papel da predação e reprodução na rede. Foi analisado que ao tomar uma valor de predação muito alto e mobilidade e reprodução baixo, no modelo estocástico a formação das espirais é comprometida. No modelo de May-Leonard quando comparadas simulações com alta taxa de reprodução e baixa taxa de mobilidade e predação, com o cenário contrario, alta taxa de predação e baixa taxa de reprodução e mobilidade, a densidade de vazios na rede se mostrá diferente. É perceptível que a densidade de vazios se encontra nas interfaces entre duas espécies.

E por fim, foi estudado a função de autocorrelação no modelo de May-Leonard. Com objetivo de investigar em como cada espécie se relaciona com ela mesmo ao longo da rede. Ao observara os gráficos é verificado como todos tem o mesmo comportamento. Eles decrescem ao longo da rede, para mobilidade alta este decréscimo acontece de maneira mais suave enquanto que para uma baixa mobilidade o decréscimo acontece de maneira mais brusca. Foi investigado também a relação da função de autocorrelação com o tamanho médio da largura de cada espiral, foi verificado que o valor da autocorrelação é aproximadamente o valor da largura das espirais.

Entre as perspectivas de trabalhos futuros, pretendemos calcular a função de autocorrelação no modelo estocástico, implementar modelos com um número de espécies maior, realizar simulação com modelo RPS bidirecional e incluir termos como canibalismo.

Código

Seguem os códigos dos programas, escritos em linguagem C, que foram utilizados para gerar os resultados mostrados durante este trabalho. Para compilar os códigos é necessário o uso das bibliotecas *math, FFTW,* e *GNU scientific Libary.*

Modelo de May-Leonard

```
#include <complex.h>
#include <fftw3.h>
#include <stdio.h>
#include <stdlib.h>
#include <gsl/gsl_rng.h>
#include <time.h>
#include <math.h>
#define Nx 500 // número de colunas da rede
#define Ny 500 // número de linhas da rede
#define dt 0.1 // passo temporal
#define p 0.1 // constante predativa
#define rp 0.8 //constante reprodutiva
#define D 0.2 // constante difusiva
#define Ns 4 // número de espécies
#define Np 200 // número de pontos
#define NG 100000 // numero de gerações*10
#define NF 2 // número de arquivos de saída

void op(double **phi, int counter); //chamada para arquivos de saída
void ic(double **phi, int seed); // chamada para início da rede
void cr(int t, double *phi); // chamada para a correlação
int main (int argc, char **argv){
int i, j, seed, counter =1;
int jp1, im1, ip1, jm1, n= 1;
int l = NG/NF;
int k, z;
double t = 0;
double p1_temp[Nx*Ny], p2_temp[Nx*Ny], p3_temp[Nx*Ny], p0_temp[Nx*Ny],
final[Nx*Ny];
double **phi;


phi = calloc(Ns, sizeof(double *));// alocação de memoria
for (i=0; i<Ns; i++){
  phi[i] = calloc((Nx*Ny), sizeof(double ));

}
if (argc == 2){
 seed= atoi(argv[1]);
}else{
```

```
 seed= time (0);
}


ic(phi, seed); //início da rede
op(phi, 0); // impressão da rede no tempo t=0
for (n=0; n<NG; n++){ // laço temporal
t = t +dt; //controle temporal

for (j=0; j < Ny; j++){ //laço espacial para as linhas durante o passo
                        //intemedário

  jp1 = (j + 1) % Ny; //definição do primeiro vizinho a direita
  jm1 = (j - 1 + Ny) % Ny; //definição do primeiro vizinho a esquerda

  for( i=0; i < Nx; i++){ //laço espacial para as colunas durante o
                          //passo intemedário

   ip1 = (i + 1) % Nx; //definição do primeiro vizinho acima
   im1 = (i - 1 + Nx) % Nx; //definição do primeiro vizinho abaixo

   //Passos intermediários para as 3 espécie
   p1_temp[i+j*Ny] = phi[1][i+j*Ny] + 0.5*dt*(rp*phi[1][i+j*Ny]*
                       phi[0][i+j*Ny] - p*phi[1][i+j*Ny]*phi[2][i+j*Ny] +
                    (phi[1][ip1+j*Ny] + phi[1][im1+j*Ny] +
                       phi[1][i+jm1*Ny] + phi[1][i+jp1*Ny] -
                       4*phi[1][i+j*Ny])*D);

   p2_temp[i+j*Ny] = phi[2][i+j*Ny] + 0.5*dt*(
                      rp*phi[2][i+j*Ny]*phi[0][i+j*Ny]
                      -p*phi[2][i+j*Ny]*phi[3][i+j*Ny] +
                    (phi[2][ip1+j*Ny] + phi[2][im1+j*Ny]+
                     phi[2][i+jm1*Ny] + phi[2][i+jp1*Ny]
                    - 4*phi[2][i+j*Ny])*D);

   p3_temp[i+j*Ny] = phi[3][i+j*Ny] + 0.5*dt*(
                      rp*phi[3][i+j*Ny]*phi[0][i+j*Ny] -
                      p*phi[3][i+j*Ny]*phi[1][i+j*Ny] +
                    (phi[3][ip1+j*Ny] + phi[3][im1+j*Ny] +
                    phi[3][i+jm1*Ny] + phi[3][i+jp1*Ny] -
                    4*phi[3][i+j*Ny])*D);

      //passo intemediário para espécie vazio
   p0_temp[i+j*Ny] = phi[0][i+j*Ny] + 0.5*dt*(
                      -rp*phi[1][i+j*Ny]*phi[0][i+j*Ny] -
                      rp*phi[2][i+j*Ny]*phi[0][i+j*Ny] -
                      rp*phi[3][i+j*Ny]*phi[0][i+j*Ny] +
                    p*phi[1][i+j*Ny]*phi[2][i+j*Ny] +
```

```
                    p*phi[2][i+j*Ny]*phi[3][i+j*Ny] +
                    p*phi[3][i+j*Ny]*phi[1][i+j*Ny] +
                    (phi[0][ip1+j*Ny] + phi[0][im1+j*Ny] +
                    phi[0][i+jp1*Ny] + phi[0][i+jm1*Ny]
                    - 4*phi[0][i+j*Ny])*D);
   }
  }

for (j=0; j<Ny; j++){ //laço espacial para as linhas durante o passo final
   jp1 = (j + 1) % Ny; //definição do primeiro vizinho a direita
   jm1 = (j - 1 + Ny) % Ny; //definição do primeiro vizinho a esquerda

   for( i=0; i < Nx; i++){ //laço espacial para as colunas durante o
                           //passo final

    ip1 = (i + 1) % Nx; // definição do primeiro vizinho acima
    im1 = (i - 1 + Nx) % Nx; //definição do primeiro vizinho abaixo

      //passo final das 3 espécies
    phi[1][i+j*Ny] = phi[1][i+j*Ny] + dt*(
                      rp*p1_temp[i+j*Ny]*p0_temp[i+j*Ny] -
                      p*p1_temp[i+j*Ny]*p2_temp[i+j*Ny] +
                   (p1_temp[ip1+j*Ny] + p1_temp[im1+j*Ny] +
                    p1_temp[i+jm1*Ny] + p1_temp[i+jp1*Ny] -
                    4*p1_temp[i+j*Ny])*D);

    phi[2][i+j*Ny] = phi[2][i+j*Ny] + dt*(
                      rp*p2_temp[i+j*Ny]*p0_temp[i+j*Ny] -
                      p*p2_temp[i+j*Ny]*p3_temp[i+j*Ny] +
                   (p2_temp[ip1+j*Ny] + p2_temp[im1+j*Ny] +
                   p2_temp[i+jm1*Ny] + p2_temp[i+jp1*Ny] -
                   4*p2_temp[i+j*Ny])*D);

    phi[3][i+j*Ny] = phi[3][i+j*Ny] + dt*(
                      rp*p3_temp[i+j*Ny]*p0_temp[i+j*Ny] -
                      p*p3_temp[i+j*Ny]*p1_temp[i+j*Ny] +
                   (p3_temp[ip1+j*Ny] + p3_temp[im1+j*Ny] +
                   p3_temp[i+jm1*Ny] + p3_temp[i+jp1*Ny] -
                   4*p3_temp[i+j*Ny])*D);

      //passo final do vazio
    phi[0][i+j*Ny] = phi[0][i+j*Ny] + dt*(
                     -rp*p0_temp[i+j*Ny]*p1_temp[i+j*Ny] -
                      rp*p0_temp[i+j*Ny]*p2_temp[i+j*Ny] -
                    rp*p0_temp[i+j*Ny]*p3_temp[i+j*Ny] +
                    p*p1_temp[i+j*Ny]*p2_temp[i+j*Ny] +
                    p*p2_temp[i+j*Ny]*p3_temp[i+j*Ny] +
                    p*p3_temp[i+j*Ny]*p1_temp[i+j*Ny] +
```

```
                    (p0_temp[ip1+j*Ny] + p0_temp[im1+j*Ny]
                    + p0_temp[i+jm1*Ny] + p0_temp[i+jp1*Ny] -
                    4*p0_temp[i+j*Ny])*D);


 }

  }
  if (((n+1)%l)== 0){ // laço para contagem de arquivos
   op(phi, counter); //saída da rede
   counter++;
   }
}
cr(1,phi[1]);// Cálculo da correlação
return 0;
}

void op(double **phi, int counter){ // saída da rede
int i ,j;
double pi;
FILE*out;
char name[64];
//geração de arquivo de saída para as três espécies 1, 2, 3
sprintf(name, "p_i-%d.dat", counter);
out= fopen(name, "w");
for (j=0; j < Ny; j++){ //laço para as colunas
 for (i=0; i < Nx; i++){ // laço para as linhas
 // impressão do valor das três espécies no sítio (i,j)
 fprintf(out, "%f ", phi[1][i+j*Ny]+2*phi[2][i+j*Ny]+3*phi[3][i+j*Ny]);
  }
fprintf(out,"\n");
 }

fclose(out);
//geração de arquvios de saída para o vazio
sprintf(name, "p_e-%d.dat", counter);
out= fopen(name, "w");
for (j=0; j < Ny; j++){//laço para as colunas
 for (i=0; i < Nx; i++){//laço para as linhas
 // impressão do valor do vazio no sítio (i,j)
 fprintf(out, "%f ", phi[0][i+j*Ny]);
  }
fprintf(out,"\n");
 }
fclose(out);
}
```

```
185 void ic(double **phi, int seed){// função para gerar início da rede
186 int i, j;
187 const gsl_rng_type * U;
188  gsl_rng * u;
189  gsl_rng_default_seed= seed;
190  U= gsl_rng_default;
191  u= gsl_rng_alloc (U); //gerador aleatório da rede
192
193
194
195 for (i = 0; i < Nx ; i++){ // laço para as colunas
196  for (j = 0; j < Ny ; j++){ //laço para as linhas
197    int S = 4 * gsl_rng_uniform (u); //geração do valor aleatório
198   switch (S){
199   case 1:
200   phi[1][i+j*Ny] = 1; // alocação para espécie 1
201   break;
202
203   case 2:
204   phi[2][i+j*Ny] = 1; // alocação para espécie 2
205   break;
206
207   case 3:
208   phi[3][i+j*Ny] = 1; // alocação para espécie 3
209   break;
210
211   case 0:
212   phi[0][i+j*Ny] = 1; // alocação para o vazio
213   break;
214    }
215  }
216 }
217
218
219 gsl_rng_free (u);
220 }
221
222 void cr(int t, double *phi){
223
224 int x, y, l, N;
225 double r;
226 double *C_r;
227 FILE *output;
228 char name[64];
229
230 C_r = calloc((Nx*Ny), sizeof(double));
231
232 // transformada de fourier do campo phi
```

```
233
234 fftw_complex *out1 = fftw_malloc(sizeof(fftw_complex)*(Nx*Ny));
235 fftw_plan FTF1= fftw_plan_dft_r2c_2d(Ny, Nx, phi, out1, FFTW_ESTIMATE);
236 fftw_execute(FTF1);
237 // calculo da funcao S(\vec{k}) a função e correlação -FTF- Transformada
238
239 for(N=0; N < Nx*Ny; N++){
240 out1[N] = conj(out1[N])*out1[N];
241 }
242
243 // inversa da funcao S(\vec{k}) -FTB- Transformada inversa
244
245 fftw_plan FTB= fftw_plan_dft_c2r_2d(Ny, Nx, out1, C_r, FFTW_ESTIMATE);
246 fftw_execute(FTB);
247
248 //calculo da função C(r) normalizada a partir da função C(\vec{r})
249
250 int norm[Np];
251 double delta_r = sqrt(Nx*Ny*2)/Np;\
252 double C[Np], mean_r[Np];
253
254 for ( l=0; l<Np; l++){
255 norm[l]= 0;
256 C[l]=0.0;
257 mean_r[l] =0.0;
258 }
259
260  for(y=0 ; y<Ny; y++){
261   for(x=0; x<Nx; x++){
262   r= sqrt((x*x)+(y*y));
263   l= (int)(r/delta_r);
264   norm[l]++;
265   mean_r[l]+= r;
266   C[l]+= C_r[y*Nx+x];
267   }
268  }
269
270 double MAX = 0.0;
271  for (l= 0; l< Np; l++){
272  mean_r[l]/= (double)(norm[l]);
273  C[l]/= (double)(norm[l]);
274   if(C[l]> MAX){
275   MAX=C[l];
276   }
277  }
278 //impressão do valor da função de Autocorrelação
279  if((output= fopen("data.dat", "w")) ==NULL){}
280  for(l=0; l<Np; l++){
```

```
 fprintf(output,"%e %e\n", mean_r[l], C[l]/MAX);
 }
 fclose(output);
//liberação de memória
 free(C_r);
 fftw_destroy_plan(FTF1);
 fftw_destroy_plan(FTB);
 fftw_free(out1);
 fftw_cleanup();
}

```

Modelo Estocástico

Para as simulações foi utilizado o seguinte código [1], foram feitas modificações no código para o mesmo se adequar as nossas simulações.

```
#include <stdio.h>
#include <time.h>
#include <gsl/gsl_rng.h>
#include <math.h>
#define Nx 500 //número de counas
#define Ny 500 // número de linhas
#define pm 0.1  // peso da predação
#define pr 0.8  // peso da reprodução
#define NG 10000 // número de gerações
#define NF 3 // número de arquivos de saída

const double pp[3][3]={\
 {-1.0,1.0,0.0},\
 {0.0,-1.0,1.0},\
 {1.0,0.0,-1.0},\
};

void op(int, int *phi); // geração de arquivos de saída
void ic(int *phi, int seed); // início da rede

int main(int argc, char **argv){
int seed;
int i, j, n, ni, nj, m, teste, ativo, passivo, vizinho, counter, n_log=22;
int l = NG/NF;
int *phi;
double p;
FILE *output;
const gsl_rng_type*W;
 gsl_rng *w;
 //gsl_rng_env_setup();
 gsl_rng_default_seed= seed;
```

```
 W = gsl_rng_default;
 w = gsl_rng_alloc (W);

 output = fopen("Ext.dat", "a");

 phi = calloc((Nx * Ny), sizeof(int)); //alocação de memória

//definições do gerador de números aleatórios
if (argc == 2){
 seed= atoi(argv[1]);
}else{
 seed= time (0);
}


 ic(phi, seed);// início da rede
 counter=1;
 for (n=0; n<NG; n++){// laço para controle das gerações
  for (m=0; m < Nx*Ny; m++){ //laço para controle de sorteios
    i= gsl_rng_uniform(w)*Nx; //sorteio da coluna para o ativo
   j= gsl_rng_uniform(w)*Ny; //sorteio da linha para o ativo
   ativo = j*Nx+i; //definição do ativo

   if(phi[ativo] != 0){
    vizinho = gsl_rng_uniform (w)*4;//sorteio do passivo
    switch (vizinho){
    case 0:
    passivo= j*Nx+(i+1+Nx)%Nx; //definição do primeiro
                               //vizinho a direita
    break;
    case 1:
    passivo= j*Nx+(i-1+Nx)%Nx; //definição do primeiro
                               //vizinho a esquerda
    break;
    case 2:
    passivo= ((j+1+Ny)%Ny)*Nx+i; //definição do primeiro vizinho acima
    break;
    case 3:
    passivo= ((j-1+Ny)%Nx)*Nx+i;// definição do primeiro
                                //vizinho abaixo
    break;
    }

    p = gsl_rng_uniform(w);
    if(p<pm){                    //ação de mobilidade
     teste = phi[ativo];
     phi[ativo]=phi[passivo];
     phi[passivo]=teste;
```

```
80      }
81      else{
82        if(p>=pm && p< (pm + pr)){
83        if(phi[passivo] ==0){   //ação de reprodução
84        phi[passivo] = phi[ativo] ;
85        }
86       }
87      else {
88        p=gsl_rng_uniform(w); //ação de predação
89       if(p<pp[phi[ativo]-1][phi[passivo]-1]){
90        phi[passivo] = 0;
91        }
92       }
93      }
94     }
95   }
96    if ((n%l)== 0){ // contagem de aquivos de saída
97     op(counter, phi);
98      counter++;
99     }
100   }
101   gsl_rng_free (w);
102   free (phi);
103   fclose(output);
104
105  return 0;
106 }
107
108 void op(int k, int *phi){ //saída da rede
109  int i, j;
110  FILE *out;
111  char nome[100];
112
113  sprintf(nome, "saida-%d.dat",k);// geração do arquivo de saida da rede
114                                  //para todas as espécies
115  out = fopen(nome, "w");
116
117  for (j=0; j<Nx; j++){ //laço para controle de colunas
118   for (i=0; i < Ny ; i++){ //laço para controle de linhas
119    fprintf( out, "%d ", phi[j*Nx+i] ); //impressão do valor
120                                        // no sítio (i,j)
121   }
122   fprintf(out,"\n");
123  }
124  fclose(out);
125 }
126
127  void ic(int *phi, int seed){ // início da rede
```

```
128  const gsl_rng_type *W;
129   //definições do gerador aleatório de números
130  gsl_rng *w;
131  gsl_rng_default_seed= seed;
132  W = gsl_rng_default;
133  w = gsl_rng_alloc (W);
134
135  int i, j, counter;
136
137  //distribuição das espécies na rede
138  for (i=1; i<4; i++){
139  counter= 1;
140  while(counter<Nx*Ny*0.33333){
141   j= gsl_rng_uniform(w)*Nx*Ny;
142   if(phi[j]== 0){
143   phi[j]=i;
144   counter++;
145   }
146  }
147  }
148
149  op(0,phi);// impressão da rede no tempo t=0
150  gsl_rng_free (w);
151  }
```

Referências Bibliográficas

[1] R. D. Bini, *Estudos da biodiversidade utilizando o jogo pedra-papel-tesoura*. Trabalho de Conclusão de Curso, Universidade Estadual de Maringá, Maringá, Parana, 2014.

[2] M. W. F. B. Kerr, M. A. Riley and B. J. M. Bohannan, "Local dispersal promotes biodiversity in a real-life game of rock-paper-scissors," *Nature*, vol. 418, p. 171, (2002).

[3] M. M. T. Reichenbach and E. Frey, "Mobility promotes and jeopardizes biodiversity in rock-paper-scissors games," *Nature*, vol. 448, p. 1046, (2007).

[4] C. Scherer, *Métodos Computacionais da Física*. Editora Livraria da Física, (2005).

[5] M. Hjorth-Jensen, *Lecture Notes on Computational Physics*. University of Oslo, (2008). Disponível em: <http://www.physics.ohio-state.edu/~ntg/780/readings/hjorth-jensen_notes2008.pdf> Acesso em: 29 Nov. 2011.

[6] E. Butkov, *Física Matemática*. LTC - Livros Técnicos e Cientifícos Editora S.A., (1988).

[7] Wikipédia, "Autocorrelação — wikipédia, a enciclopédia livre," (2014). [Online; accessed 10-fevereiro-2016].

[8] P. P. Avelino, D. Bazeia, L. Losano, J. Menezes, and B. F. Oliveira, "Junctions and spiral patterns in generalized rock-paper-scissors models," *Phys. Rev. E*, vol. 86, p. 036112, Sep (2012).

[9] P. Avelino, D. Bazeia, J. Menezes, and B. de Oliveira, "String networks in lotka–volterra competition models," *Physics Letters A*, vol. 378, no. 4, pp. 393 – 397, (2014).

[10] Wikipédia, "Vizinhança de von neumann — wikipédia, a enciclopédia livre," (2015). [Online; accessed 10-fevereiro-2016].

[11] P. P. Avelino, D. Bazeia, L. Losano, J. Menezes, and B. F. de Oliveira, "Interfaces with internal structures in generalized rock-paper-scissors models," *Phys. Rev. E*, vol. 89, p. 042710, Apr (2014).