Modelação Matemática de uma Concha do Náutilus usando Python/turtle

Adrian Dias
Número 1

October 3, 2025

Abstract

Este relatório apresenta o desenvolvimento de um modelo matemático e a respectiva implementação em `Python/turtle`, com vista à representação de uma concha do náutilus encontrada na natureza. São descritos os objectivos, a metodologia seguida e os principais resultados obtidos.

1 Introdução

Neste capítulo apresenta-se o enquadramento teórico do trabalho, incluindo o modelo matemático de base e o modelo específico implementado no código.

1.1 Modelo Matemático da Espiral Logarítmica (1.1)

A concha do náutilus segue uma espiral logarítmica, também conhecida como espiral de crescimento ou espiral equiangular. Esta curva é representada pela equação polar:

$$r(\theta) = a \cdot e^{b \cdot \theta} \tag{1}$$

onde:

- r é o raio da espiral (distância do ponto à origem)
- θ é o ângulo em radianos (parâmetro angular)
- a é o raio inicial (constante de escala que determina o tamanho inicial)
- b é a taxa de crescimento (controla a "abertura" ou "fechamento" da espiral)
- e é a base do logaritmo natural ($e \approx 2.71828$)

1.2 Propriedades Matemáticas Fundamentais (1.2)

A espiral logarítmica possui propriedades matemáticas notáveis que explicam sua prevalência na natureza:

- **Auto-similaridade**: A forma mantém-se invariante sob transformações de escala. Isto significa que qualquer secção da espiral é geometricamente similar à espiral completa.
- **Ângulo constante**: O ângulo α entre o raio vetor e a tangente à curva é constante em todos os pontos, satisfazendo a relação $\cot(\alpha) = b$.
- **Crescimento exponencial**: O raio cresce exponencialmente com o ângulo, o que corresponde a uma taxa de crescimento proporcional ao tamanho atual.
- **Propriedade de crescimento isomórfico**: A forma da espiral mantém-se constante durante o crescimento, o que é energeticamente eficiente para organismos vivos.

1.3 Implementação Computacional (1.3)

No script `Python/turtle`, a equação 1 é implementada através da conversão de coordenadas polares para cartesianas:

$$x = r(\theta) \cdot \cos(\theta), \quad y = r(\theta) \cdot \sin(\theta) \tag{2}$$

Para criar a forma tridimensional da concha, utilizou-se o conceito de faixa espiral, onde uma segunda espiral com raio $r_2(\theta) = r_1(\theta) + d(\theta)$ é desenhada, sendo $d(\theta)$ uma função que define a largura da concha.

1.4 Formulação por Equações Diferenciais (1.4)

A espiral logarítmica surge naturalmente de sistemas dinâmicos. Considerando o sistema:

$$\begin{cases} \dot{x} - \dot{y} = \ddot{x}\sqrt{\dot{x}^2 + \dot{y}^2} \\ \dot{x} + \dot{y} = \ddot{y}\sqrt{\dot{x}^2 + \dot{y}^2} \end{cases} \tag{3}$$

A solução em coordenadas polares conduz ao sistema:

$$\dot{r} = 1, \quad \dot{\theta} = \frac{1}{r} \tag{4}$$

cuja solução é a espiral logarítmica $r = c_0 \cdot e^{\theta}$, demonstrando como padrões naturais emergem de leis dinâmicas simples.

1.5 Propriedade de Inversão (1.5)

A inversão da espiral logarítmica no círculo unitário, dada por $r_{inv}(\theta) = 1/r(\theta)$, produz outra espiral logarítmica. Esta propriedade é particularmente relevante em transformações geométricas e mapeamentos conformes.

1.6 Cálculo de Áreas na Concha (1.6)

A área de uma secção da concha pode ser calculada através da integral:

$$A = \frac{1}{2}\int_{\theta_1}^{\theta_2} r^2 d\theta = \frac{1}{2}\int_{\theta_1}^{\theta_2} a^2 e^{2b\theta} d\theta = \frac{a^2}{4b}(e^{2b\theta_2} - e^{2b\theta_1}) \tag{5}$$

Para a concha do náutilus, consideramos normalmente $\theta_1 = 6\pi$ e $\theta_2 = 8\pi$ para a última revolução, resultando numa área que cresce exponencialmente com cada volta.

1.7 Modelo de Faixa Espiral para Conchas (1.7)

Para representar a concha 3D, utilizou-se o modelo de faixa espiral com duas espirais concêntricas:

$$\begin{cases} r_1(\theta) = a \cdot e^{b\theta} & \text{(espiral interna)} \\ r_2(\theta) = r_1(\theta) + d(\theta) & \text{(espiral externa)} \end{cases} \tag{6}$$

onde $d(\theta)$ representa a largura da concha. A área da secção transversal da última revolução é dada por:

$$A_{seccao} = \frac{1}{2}\int_{6\pi}^{8\pi} [r_2^2(\theta) - r_1^2(\theta)] d\theta \tag{7}$$

No caso de uma concha real, $d(\theta)$ tipicamente aumenta com θ para acomodar o crescimento do organismo.

1.8 Parâmetros e Ajuste

Os parâmetros a e b foram ajustados empiricamente para aproximar a forma de uma concha real:

- $a = 2.0$: Define o tamanho inicial da concha
- $b = 0.22$: Controla a taxa de abertura da espiral
- $\theta \in [0, 8\pi]$: Permite 4 voltas completas da espiral
- $d(\theta) = 0.4 \cdot r(\theta)$: Largura proporcional ao raio atual

Figure 1: Concha do Náutilus - modelo matemático implementado em Python/turtle.

2 Parte Experimental

A componente experimental deste trabalho corresponde à elaboração e explicação do código em `Python` com a biblioteca `turtle`. Tal como num procedimento laboratorial, importa detalhar a lógica implementada, os algoritmos utilizados e as opções tomadas em cada etapa, de forma a permitir a replicação do processo.

2.1 Arquitetura do Código

O script foi estruturado em funções modulares para facilitar a compreensão e manutenção:

- `espiral_logaritmica(a, b, theta)`: Calcula pontos na espiral
- `desenhar_concha_realista()`: Função principal de desenho
- `desenhar_camaras(pontos)`: Adiciona divisórias internas
- `desenhar_estrias()`: Cria linhas de crescimento

2.2 Implementação da Espiral Logarítmica

A equação fundamental $r(\theta) = a \cdot e^{b\theta}$ foi implementada da seguinte forma:

```
def espiral_logaritmica(a, b, theta):
    r = a * math.exp(b * theta)
    x = r * math.cos(theta)
    y = r * math.sin(theta)
    return x, y, r
```

2.3 Conversão Coordenadas Polares-Cartesianas

Para compatibilidade com o sistema `turtle`, as coordenadas polares são convertidas:

$$x = r \cdot \cos(\theta), \quad y = r \cdot \sin(\theta) \tag{8}$$

O incremento angular $\Delta\theta = 0.01$ radianos garante suavidade na curva.

2.4 Modelo de Faixa Espiral

Para criar a espessura da concha, implementou-se o conceito de faixa espiral com duas curvas:

```
# Espiral interna
r1 = a * math.exp(b * theta)
# Espiral externa
r2 = r1 + 0.4 * r1  # Largura proporcional
```

2.5 Parâmetros Ajustados

Através de testes iterativos, determinou-se os parâmetros ótimos:

- **a = 2.0**: Tamanho inicial que equilibra visualização e detalhe
- **b = 0.22**: Taxa de crescimento que produz forma realista
- **Ângulo total**: 8π radianos (4 voltas completas)
- **Resolução**: 800 pontos para suavidade adequada

2.6 Algoritmo de Desenho

O desenho segue a sequência:

1. Inicialização do ambiente `turtle`
2. Desenho da espiral interna (do centro para fora)
3. Desenho da espiral externa (retorno)
4. Preenchimento da área entre espirais
5. Adição de elementos detalhados (câmaras, estrias)

2.7 Otimizações Implementadas

Para melhor performance e qualidade visual:

- **Velocidade máxima**: `turtle.speed(0)`
- **Preenchimento inteligente**: Uso de `begin_fill()/end_fill()`
- **Ocultação do cursor**: `turtle.hideturtle()`
- **Gestão de cores**: Gradientes para realismo

2.8 Tratamento de Erros

O código inclui verificações para:

- Existência de diretórios de saída
- Parâmetros dentro de intervalos válidos
- Finalização graciosa com `screen.bye()`

2.9 Metodologia de Testes

O desenvolvimento seguiu uma abordagem iterativa:

1. Implementação básica da espiral simples
2. Adição progressiva de complexidade (faixa espiral)
3. Refinamento visual (cores, texturas)
4. Otimização de parâmetros
5. Validação contra referências visuais

Este processo permitiu alcançar um equilíbrio entre fidelidade matemática e apelo visual, resultando numa representação convincente da concha do náutilus.

3 Discussão dos Resultados

Apresentam-se e analisam-se, nesta secção, as imagens geradas automaticamente pelo código. Não foram utilizadas capturas de ecrã, mas sim exportações directas produzidas pelo programa. Discutem-se as semelhanças e diferenças entre os resultados e a imagem de referência, identificando as causas dos desvios e avaliando a qualidade da aproximação obtida.

4 Conclusões

As conclusões são redigidas a partir da análise dos resultados. Evitam-se afirmações superficiais ou subjectivas; privilegiam-se observações fundamentadas, como, por exemplo:

- O modelo reproduz com fidelidade parcial a forma natural seleccionada.
- As limitações decorrem de aproximações matemáticas ou restrições do ambiente de programação.
- Futuras melhorias poderão incluir optimizações algorítmicas ou refinamentos gráficos.

5 Bibliografia

References

[1] Carvalho, J. (2021). *Práticas de Programação em Python*. Editora XYZ.

[2] Martins, A. e Silva, M. (2015). *Programação Científica com Python*. Editora ABC.

[3] Math Stack Exchange. (2019). *Deriving the Nautilus shell spiral equation.*