UNIOESTE – Universidade Estadual do Oeste do Paraná

CECE– Centro de Engenharias e Ciências Exatas

Curso de Ciência da computação

Disciplina: Cálculo Numérico

**FENÔMENO DE RUNGE**

Isabela Pimentel Loebel

FOZ DO IGUAÇU

2018

Pontos:

(-5), (-3.88889), (-2.77778), (-1.66667), (-0.555556), 0.555556, 1.66667, 2.77778, 3.88889, 5, sendo a diferença entre os pontos de 1.11111.

Função:

A função é F(x)=1/(1+25x2)e seu gráfico é:

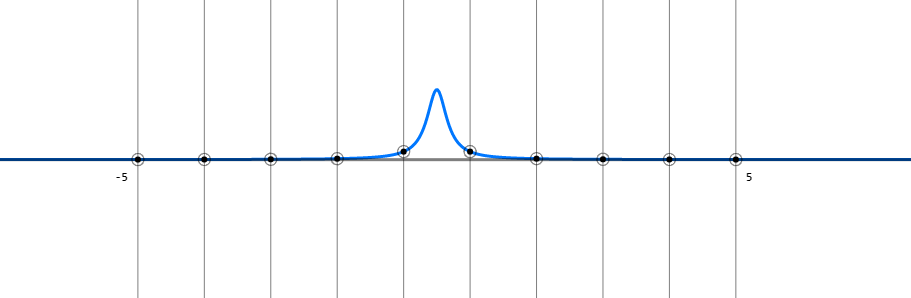


Gráfico 1 - gráfico da função no intervalo [-5, 5].

Polinômio:

0.135472 + (0\*x) + (-0.0707116\*x^2) + (0\*x^3) + (0.0115895\*x^4) + (1.48685\*10^-18\*x^5) + (-7.04661\*10^-4\*x^6) + (-1.46239\*10^-19\*x^7) + (1.38261\*10^-5\*x^8) + (3.63798\*10^-21\*x^9)

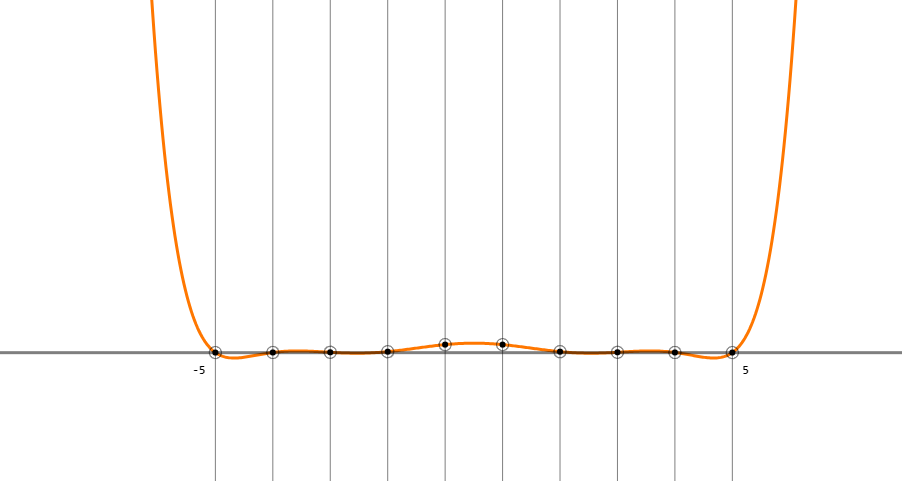


Gráfico 2 - gráfico do polinômio no intervalo [-5, 5].

Splines:

[-5, -3.88889] = 9.36429\*10^-4\*x + 0.00627959

(-3.88889, -2.77778] = 0.00226741\*x + 0.0114556

(-2.77778, -1.66667] = 0.00813449\*x + 0.0277531

(-1.66667, -0.555556] = 0.0904818\*x + 0.164999

(-0.555556, 0.555556] = 0\*x + 0.114731

(0.555556, 1.66667] = -0.0904818\*x + 0.164999

(1.66667, 2.77778] = -0.00813449\*x + 0.0277531

(2.77778, 3.88889] = -0.00226741\*x + 0.0114556

(3.88889, 5] = -9.36429\*10^-4\*x + 0.00627959

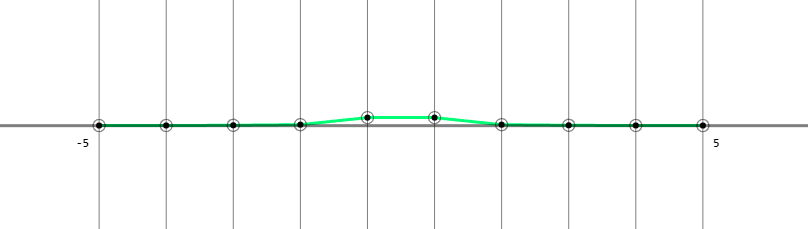


Gráfico 3 - gráfico das splines no intervalo [-5, 5].

Código fonte do site usado:

OBS: devido ao código fonte do site usado ser muito extenso e também por ele realizar gráficos, foi necessário pegar somente a parte central do código, não constando ele inteiro e também o uso de colunas, mas sem alterações no código original.

(...)

var functions = [

{

f: x=> NaN,

color: "#f70",

name: "Polinômio",

render: 1

},

{

f: x=> 1/(1 + 25\*x\*x),

color: "#07f",

name: "1 / (1 + 25x^2)",

render: 1

},

{

f: x=> NaN,

color: "#0f7",

name: "Spline",

render: 1

},

{

f: x=> 0,

color: "rgba(255,255,255,0.5)",

render: 1

}

];

var valToPxX = val => (val - val\_x0) / (val\_x1 - val\_x0) \* screenSx;

var pxToValX = px => (px / screenSx) \* (val\_x1 - val\_x0) + val\_x0;

var valToPxY = val => screenSy - (val - val\_y0) / (val\_y1 - val\_y0) \* screenSy;

var pxToValY = px => val\_y0 - (px - screenSy) / screenSy \* (val\_y1 - val\_y0);

function stopCentralizeAnimation() {

if (animation\_interval != null) {

clearInterval(animation\_interval);

}

}

function centralize() {

var ax = Math.min(val\_ax, val\_bx);

var bx = Math.max(val\_ax, val\_bx);

var twidth = val\_x1 - val\_x0;

var graphArea = bx - ax;

var space = (twidth - graphArea) \* 0.5;

var prev\_val\_x0 = val\_x0;

var prev\_val\_x1 = val\_x1;

var new\_val\_x0 = ax - space;

var new\_val\_x1 = bx + space;

stopCentralizeAnimation();

var ini = new Date();

var time = 2000;

animation\_interval = setInterval(function(){

var now = new Date();

var t = (now - ini) / time;

if (t >= 1) {

t = 1;

stopCentralizeAnimation();

}

var a = (1-Math.cos(t\*Math.PI))\*0.5;

var b = 1 - a;

val\_x0 = prev\_val\_x0 \* b + new\_val\_x0 \* a;

val\_x1 = prev\_val\_x1 \* b + new\_val\_x1 \* a;

updateCanvas();

}, 0);

}

function updateConsts() {

if (last\_ax === val\_ax && last\_bx === val\_bx && last\_n === n && last\_sn === sciNote) {

return;

}

last\_ax = val\_ax;

last\_bx = val\_bx;

last\_n = n;

last\_sn = sciNote;

var inix = Math.min(val\_bx, val\_ax);

var endx = Math.max(val\_bx, val\_ax);

var interval = (endx - inix) / (n - 1);

var f = functions[1].f;

for (var i=0; i<n; ++i) {

vx[i] = inix + i\*interval;

vy[i] = f(vx[i]);

}

for (var row=n; row--;) {

var x = 1;

var mul = vx[row];

for (var col=n; col--;) {

matrix[row][col] = x;

x \*= mul;

}

}

for (var i=n; i--; matrix[i][n] = vy[i]);

for (var i=0; i<n; ++i) {

var max = Math.abs(matrix[i][i]);

var pos = i;

for (var j=i+1; j<n; ++j) {

var val = Math.abs(matrix[j][i]);

if (val > max) {

max = val;

pos = j;

}

}

if (pos != i) {

var line = matrix[i];

matrix[i] = matrix[pos];

matrix[pos] = line;

}

}

for (var a=1; a<n; ++a) {

var x = a - 1;

for (var b=a; b<n; ++b) {

var e = matrix[b][x];

var s = matrix[x][x];

var m = e / s;

matrix[b][x] = 0;

for (var i=x+1; i<=n; ++i) {

matrix[b][i] -= matrix[x][i] \* m;

}}}

for (var i=n; i--;) {

var equals = matrix[i][n];

for (var j=n; --j>i;) {

equals -= matrix[i][j] \* consts[j];

}

consts[i] = equals / matrix[i][i];

}

var f1\_text = "";

var f1\_eval = "functions[0].f=x=>";

var xtext = "\*x";

for (var i=n, p=0; i--; ++p) {

if (f1\_text) {

f1\_text += " + ";

f1\_eval += "+";

}

f1\_text += sciNote(consts[i]);

f1\_eval += consts[i];

if (p > 1) {

f1\_text += "\*x^" + p;

xtext += "\*x";

f1\_eval += xtext;

} else if (p === 1) {

f1\_text += "\*x";

f1\_eval += "\*x";

}}

if (val\_ax != null) {

functions[0].lines = [f1\_text];

var f2\_lines = [];

functions[2].lines = f2\_lines;

var f2\_eval = "functions[2].f=x=>x<" + vx[0] + "?NaN";

for (var i=1; i<n; ++i) {

var a = vx[i-1];

var b = vx[i];

var str;

if (i === 1) {

str = "[";

} else {

str = "(";

}

str += sciNote(a) + ", " + sciNote(b) + "] = ";

var dx = b - a;

var dy = vy[i] - vy[i-1];

var m = dy / dx;

str += sciNote(m) + "\*x + " + sciNote(vy[i-1] - vx[i-1] \* m);

f2\_lines.push(str);

f2\_eval += ":x<=" + b + "?x\*" + m + " + " + (vy[i-1] - vx[i-1] \* m);

}

eval(f1\_eval);

eval(f2\_eval + ":NaN;");

}}

var dom\_panel = document.querySelector(".panel");

var panel\_style = window.getComputedStyle(dom\_panel);

function updateTexts() {

var contents = document.querySelectorAll(".content");

var obj = functions[0];

contents[0].innerText = obj.lines && obj.lines.length ? obj.lines.join("\n") : "";

obj = functions[2];

contents[1].innerText = obj.lines && obj.lines.length ? obj.lines.join("\n") : "";

if (points.length && !isNaN(points[0])) {

contents[2].innerText = points.join(", ");

}}

function updateCanvas() {

var lastSx = screenSx;

var lastSy = screenSy;

var panelSx = parseInt(panel\_style.width);

panelSx += parseInt(panel\_style.paddingRight);

panelSx += parseInt(panel\_style.paddingLeft);

screenSx = window.innerWidth - panelSx;

screenSy = window.innerHeight;

if (lastSx !== screenSx || lastSy !== screenSy) {

canvas.setAttribute("width", screenSx);

canvas.setAttribute("height", screenSy);

}

updateConsts();

ctx.lineWidth = 3;

ctx.lineJoin = "round";

ctx.lineCap = "round";

ctx.fillStyle = background\_color;

ctx.fillRect(0, 0, screenSx, screenSy);

for (var i=0; i<functions.length; ++i) {

var obj = functions[i];

if (!obj.render) {

continue;

}

var f = obj.f;

ctx.strokeStyle = functions[i].color;

ctx.beginPath();

var dx = val\_x1 - val\_x0;

var dy = val\_y1 - val\_y0;

var const\_1 = dx / screenSx;

var const\_2 = 1 / (dy \* screenSy);

var last\_y = NaN;

for (var px=0; px<=screenSx; px+=step\_x) {

var x = pxToValX(px);

var y = f(x);

if (y !== NaN) {

if (last\_y === NaN) {

ctx.moveTo(px, valToPxY(y));

} else {

ctx.lineTo(px, valToPxY(y));

}}

last\_y = y;

}

ctx.stroke();

ctx.closePath();

}

var tx = valToPxX(val\_test);

var ax = valToPxX(val\_ax);

var bx = valToPxX(val\_bx);

var interval = (bx - ax) / (n - 1);

ctx.strokeStyle = lineColor;

ctx.lineWidth = 0.5;

ctx.beginPath();

for (var i=0; i<n; ++i) {

ctx.moveTo(ax + i\*interval, 0);

ctx.lineTo(ax + i\*interval, screenSy);

}

if (val\_test != null) {

ctx.moveTo(tx, 0);

ctx.lineTo(tx, screenSy);

}

ctx.stroke();

ctx.closePath();

var fontSize = 12;

var margin = 10;

var space = 5;

var dy = fontSize + space;

ctx.font = "bold " + fontSize + "px Monospace";

var val\_a = val\_ax;

var val\_b = val\_bx;

if (ax > bx) {

var aux = ax;

ax = bx;

bx = aux;

aux = val\_a;

val\_a = val\_b;

val\_b = aux;

}

ctx.fillStyle = lineColor;

ctx.textBaseline = "top";

ctx.textAlign = "right";

var origin = valToPxY(0);

ctx.fillText(sciNote(val\_a), ax - margin, origin + margin);

ctx.textAlign = "left";

ctx.fillText(sciNote(val\_b), bx + margin, origin + margin);

if (val\_test != null) {

ctx.fillText(sciNote(val\_test), tx + margin, margin);

}

ctx.textBaseline = "bottom";

ctx.fillText(n + " pontos", margin, screenSy - margin);

if (val\_test != null && val\_a != null && val\_b != null) {

var y = screenSy - (dy + margin);

for (var i=0; i<functions.length; ++i) {

var obj = functions[i];

if (obj.render && obj.name) {

var val = obj.f(val\_test);

ctx.fillStyle = obj.color;

if (isNaN(val)) {

ctx.fillText(obj.name, margin, y);

} else {

ctx.fillText(obj.name + ": " + sciNote(val), margin, y);

}

y -= dy;

}}}

var y = valToPxY(0);

ctx.strokeStyle = ctx.fillStyle = white ? "#000" : "#fff";

var inix = Math.min(val\_ax, val\_bx);

var endx = Math.max(val\_ax, val\_bx);

var interval = (endx - inix) / (n - 1);

var f = functions[1].f;

points = [];

for (var i=0; i<n; ++i) {

var x = i \* interval + inix;

points.push(sciNote(x));

var px\_y = valToPxY(f(x));

var px\_x = valToPxX(x);

ctx.beginPath();

ctx.arc(px\_x, px\_y, point\_rad, 0, Math.PI\*2);

ctx.fill();

ctx.closePath();

ctx.beginPath();

ctx.arc(px\_x, px\_y, point\_rad \* 2, 0, Math.PI\*2);

ctx.stroke();

ctx.closePath();

}

updateTexts();

}

var mouseIsDown = false;

canvas.addEventListener("mousedown", function(event){

if (event.button === 0) {

stopCentralizeAnimation();

mouseIsDown = true;

val\_bx = val\_ax = pxToValX(event.offsetX);

updateCanvas();

}});

canvas.addEventListener("mouseup", function(event){

mouseIsDown = false;

centralize();

});

canvas.addEventListener("mousemove", function(event){

if (!(event.buttons & 1)) {

if (mouseIsDown) {

centralize();

}

mouseIsDown = false;

}

if (mouseIsDown) {

val\_bx = pxToValX(event.offsetX);

}

if (!white) {

val\_test = pxToValX(event.offsetX);

}

updateCanvas();

});

canvas.addEventListener("wheel", function(event){

stopCentralizeAnimation();

event.preventDefault();

var slice = (val\_x1 - val\_x0) \* event.deltaY \* 0.001;

val\_x0 -= slice;

val\_x1 += slice;

updateCanvas();

});

(…)

Referências:

<http://giovanirubim.herokuapp.com/runge/>