f(x) = a\*x\*b\*\*(1-x)

f‘(x)= -a \* (ln(b)\*x-1) \* b\*\*(1-x)

Nullstelle f’(x0) = 0 => x0=1/ln(b)

f‘(x0) = (a/ln(b))\*b\*\*(1-1/ln(b))

Nullstelle vorgegeben (x0/y0) =>

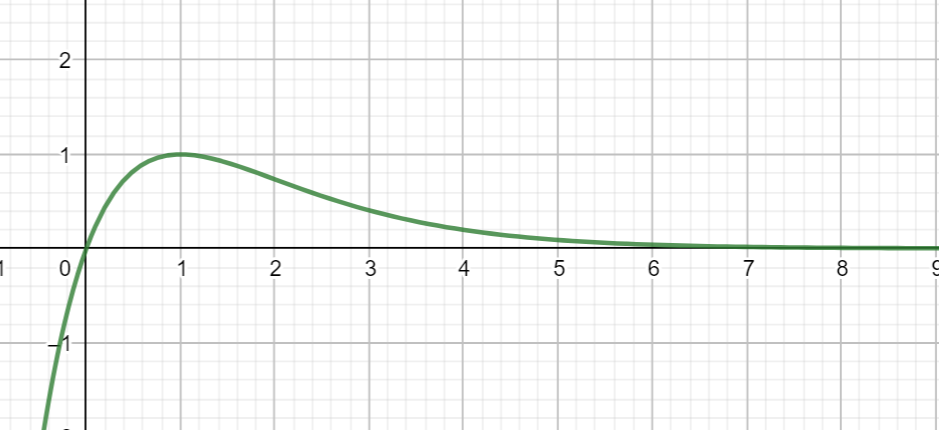
a = y0 / (x0 \* (e\*\*(1/x0-1)))

b = e\*\*(1/x0)

r := 1/x0

a = y0/ (x0 \* (e\*\*(r-1))) = (y0/x0) \* e\*\*(1-r)

b = e\*\*r



f(x) = x \* e hoch(1-x)

f(x) = a\*x\*b\*\*(1-x)

a = (y0/x0) \* e\*\*(1-r)

b = e\*\*r

r := 1/x0

q:= x /x0

f(x) = y0 \* e\*\*(1-q) \*q

//fPixelPoint const StimulusDesigner::getIntegralPnt(fMicroSecs const time) const

//{

// float const fY { m\_pSignalGenerator->StimulusIntegral(time) };

// return getPixPnt(time, fHertz(fY));

//}

//float const SignalGenerator::StimulusIntegral(fMicroSecs const stimulusTime) const

//{

// fMicroSecs const time { (stimulusTime < m\_usCutoffTime) ? stimulusTime : m\_usCutoffTime };

// float const x { time.GetValue() \* 1e-3f }; // convert to milliseconds

// float const lnB { log(m\_fParamB) };

// float const C { m\_fParamA.GetValue() \* m\_fParamB/(lnB\*lnB) };

// float const fRes { C - m\_fParamA.GetValue() \* (lnB \* x + 1) \* pow(m\_fParamB, (1.0f - x))/(lnB\*lnB) };

// return fRes \* 1e-3f;

//}

//prevPoint = getIntegralPnt(timeStart);

//for (fMicroSecs time = timeStart + usIncrement; time < timeEnd; time += usIncrement)

//{

// fPixelPoint const actPoint { getIntegralPnt(time) };

// fPixelPoint const stepPoint { actPoint.GetX(), prevPoint.GetY() };

// m\_graphics.DrawLine(prevPoint, stepPoint, m\_fPixLineWidth, D2D1::ColorF::Red);

// m\_graphics.DrawLine(stepPoint, actPoint, m\_fPixLineWidth, D2D1::ColorF::Red);

// prevPoint = actPoint;

//}