{ "cells": [ { "cell_type": "code", "execution_count": 36, "id": "54e3a357-7469-4ba5-b14e-f6711c1f6b7d", "metadata": {}, "outputs": [ { "data": { "image/png": "iVBORw0KGgoAAAANSUhEUgAAAagAAAGoCAYAAAATsnHAAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjUuMCwgaHR0cHM6Ly9tYXRwbG90b
"text/plain": [ "

" ] }, "metadata": { "needs_background": "light" }, "output_type": "display_data" } ], "source": [ "#Solve the non forced using RK4\n", "#here F(t) = 0, therefore we have a homogenous linear equations\n", "\n", "# Overdamped case where zeta > 1:\n", "\n", "%matplotlib inline\n", "import numpy as np\n", "from math import*\n", "import matplotlib.pyplot as plt\n", "import matplotlib.gridspec as gridspec\n", "\n", "c1 = 1./6\n", "c2 = 2./6\n", "c3 = 2./6\n", "c4 = 1./6\n", "a2 = 1./2\n", "a3 = 1./2\n", "a4 = 1.\n", "b21 = 1./2\n", "b31 = 0.\n", "b32 = 1./2\n", "b41 = 0.\n", "b42 = 0.\n", "b43 = 1.\n", "omega_n = 1.\n", "zeta = 1.3\n", "\n", "lambda1 = -omega_n * (zeta - sqrt(zeta**2 - 1)) #first solution\n", "lambda2 = -omega_n * (zeta + sqrt(zeta**2 - 1)) #Second solution\n", "\n", "\n", "fxt = lambda x, t: (lambda1/(lambda1-lambda2))*exp(lambda1*t)+(lambda2/(lambda2-lambda1))*exp(lambda2*t)\n", "\n", "def xt_overdamped(sol1, sol2, t):\n", " return (1/(sol1-sol2))*exp(sol1*t)+(1/(sol2-sol1))*exp(sol2*t)\n", "def vt(sol1,sol2,t):\n", " return sol1/(lI-lII)*exp(lI*t)+lII/(lII-lI)*exp(lII*t)\n", "\n", "ti = 0\n", "tf = 10\n", "n = 100\n", "h = (tf-ti)/(n-1)\n", "t = np.linspace(ti, tf, n)\n", "x_overdamped = [xt_overdamped(lambda1, lambda2,tval) for tval in t]\n", "\n", "\n", "iv = 0.8\n", "x_overdamped[0] = iv \n", " \n", "def rk4(ti, xi, h):\n", " K1 = fxt(ti,xi)\n", " K2 = fxt(ti+a2*h,xi+b21*K1*h)\n", " K3 = fxt(ti+a3*h,ti+b31*K1*h+b32*K2*h)\n", " K4 = fxt(ti+a4*h,ti+b41*K1*h+b42*K2*h+b43*K3*h) \n", " xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*h\n", " return xip1\n", "\n", " \n", "for i in range(1,n):\n", " t[i] = ti + i*h\n", " x_overdamped[i] = rk4(t[i-1],x_overdamped[i-1],h)\n", " \n", "fig = plt.figure(figsize=(6,6))\n", "plt.plot(t, x_overdamped, label=\"Runge-Kutta 4\",color=\"r\",linewidth=\"2.0\")\n", "plt.legend()\n", "plt.tight_layout()\n", "plt.show()" ] }, { "cell_type": "code", "execution_count": 26, "id": "a9cd74fc-2c7a-4be1-8794-7b3eb0dca294", "metadata": { "tags": [] }, "outputs": [ { "data": { "image/png": "iVBORw0KGgoAAAANSUhEUgAAAagAAAGoCAYAAAATsnHAAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjUuMCwgaHR0cHM6Ly9tYXRwb
"text/plain": [ "

" ] }, "metadata": { "needs_background": "light" }, "output_type": "display_data" } ], "source": [ "#Solve the non forced using RK4\n", "#here F(t) = 0, therefore we have a homogenous linear equations\n", "\n", "# Critically damped case where zeta = 1:\n", "\n", "%matplotlib inline\n", "import numpy as np\n", "from math import*\n", "import matplotlib.pyplot as plt\n", "import matplotlib.gridspec as gridspec\n", "\n", "c1 = 1./6\n", "c2 = 2./6\n", "c3 = 2./6\n", "c4 = 1./6\n", "a2 = 1./2\n", "a3 = 1./2\n", "a4 = 1.\n", "b21 = 1./2\n", "b31 = 0.\n", "b32 = 1./2\n", "b41 = 0.\n", "b42 = 0.\n", "b43 = 1.\n", "omega_n = 1.\n", "\n", "\n", "fxt = lambda x, t: exp(-omega_n*t)-t*omega_n*exp(-omega_n*t)\n", "\n", "def xt_critdamped(omega_n, t):\n", " return t*exp(-omega_n*t)\n", "\n", "ti = 0\n", "tf = 30\n", "n = 100\n", "h = (tf-ti)/(n-1)\n", "t = np.linspace(ti, tf, n)\n", "x_critdamped = [xt_critdamped(4.,tval) for tval in t]\n", "\n", "iv = 0.8\n", "x_critdamped[0] = iv \n", " \n", "def rk4(ti, xi, h):\n", " K1 = fxt(ti,xi)\n", " K2 = fxt(ti+a2*h,xi+b21*K1*h)\n", " K3 = fxt(ti+a3*h,ti+b31*K1*h+b32*K2*h)\n", " K4 = fxt(ti+a4*h,ti+b41*K1*h+b42*K2*h+b43*K3*h) \n", " xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*h\n", " return xip1\n", "\n", " \n", "for i in range(1,n):\n", " t[i] = ti + i*h\n", " x_critdamped[i] = rk4(t[i-1],x_critdamped[i-1],h) \n", " \n", "fig = plt.figure(figsize=(6,6))\n", "plt.plot(t, x_critdamped, label=\"Runge-Kutta 4\",color=\"b\",linewidth=\"2.0\")\n", "plt.legend()\n", "plt.tight_layout()\n", "plt.show()" ] }, { "cell_type": "code", "execution_count": 35, "id": "5c9736ed-7694-4a2f-b18a-115e1fb33672", "metadata": {}, "outputs": [ { "data": { "image/png": "iVBORw0KGgoAAAANSUhEUgAAAagAAAGoCAYAAAATsnHAAAAAOXRFWHRTb2Z0d2FyZQBNYXRwbG90bGliIHZlcnNpb24zLjUuMCwgaHR0cHM6Ly9tY
"text/plain": [ "

" ] }, "metadata": { "needs_background": "light" }, "output_type": "display_data" } ], "source": [ "#Solve the non forced using RK4\n", "#here F(t) = 0, therefore we have a homogenous linear equations\n", "\n", "# Overdamped case where zeta > 1:\n", "\n", "%matplotlib inline\n", "import numpy as np\n", "from math import*\n", "import matplotlib.pyplot as plt\n", "import matplotlib.gridspec as gridspec\n", "\n", "c1 = 1./6\n", "c2 = 2./6\n", "c3 = 2./6\n", "c4 = 1./6\n", "a2 = 1./2\n", "a3 = 1./2\n", "a4 = 1.\n", "b21 = 1./2\n", "b31 = 0.\n", "b32 = 1./2\n", "b41 = 0.\n", "b42 = 0.\n", "b43 = 1.\n", "omega_n = 4\n", "zeta = 0.01\n", "omega_d = omega_n*sqrt(1-zeta**2)\n", "\n", "fxt = lambda x, t: exp(-zeta*omega_n*t)*((cos(omega_d*t)+zeta*pow(omega_n, 2)/omega_d*sin(omega_d*t)) + (-omega_d*sin(omega_d*t)+zeta*omega_n*cos(omega_d*t)))\n", "\n", "def xt_underdamped(zeta,omega_n,omega_d,t):\n", " return exp(-zeta*omega_n*t)*(cos(omega_d*t)+zeta*omega_n/omega_d*sin(omega_d*t))\n", " \n", "\n", "ti = 0\n", "tf = 10.\n", "n = 500\n", "h = (tf-ti)/(n-1)\n", "t = np.linspace(ti, tf, n)\n", "x_underdamped = [xt_underdamped(zeta,omega_n,omega_d,tval) for tval in t]\n", "\n", "\n", "iv = 0.8\n", "x_underdamped[0] = iv \n", " \n", "def rk4(ti, xi, h):\n", " K1 = fxt(ti,xi)\n", " K2 = fxt(ti+a2*h,xi+b21*K1*h)\n", " K3 = fxt(ti+a3*h,ti+b31*K1*h+b32*K2*h)\n", " K4 = fxt(ti+a4*h,ti+b41*K1*h+b42*K2*h+b43*K3*h) \n", " xip1 = xi+(c1*K1+c2*K2+c3*K3+c4*K4)*h\n", " return xip1\n", "\n", " \n", "for i in range(1,n):\n", " t[i] = ti + i*h\n", " x_underdamped[i] = rk4(t[i-1],x_underdamped[i-1],h)\n", " \n", "fig = plt.figure(figsize=(6,6))\n", "plt.plot(t, x_underdamped, label=\"Runge-Kutta 4\",color=\"g\",linewidth=\"2.0\")\n", "plt.legend()\n", "plt.tight_layout()\n", "plt.show()" ] }, { "cell_type": "code", "execution_count": null, "id": "81450abe-5fbf-40fa-9ea3-a480ef9b2789", "metadata": {}, "outputs": [], "source": [] } ], "metadata": { "kernelspec": { "display_name": "Python 3 (ipykernel)", "language": "python", "name": "python3" }, "language_info": { "codemirror_mode": { "name": "ipython", "version": 3 }, "file_extension": ".py", "mimetype": "text/x-python", "name": "python", "nbconvert_exporter": "python", "pygments_lexer": "ipython3", "version": "3.9.5" } }, "nbformat": 4, "nbformat_minor": 5 }