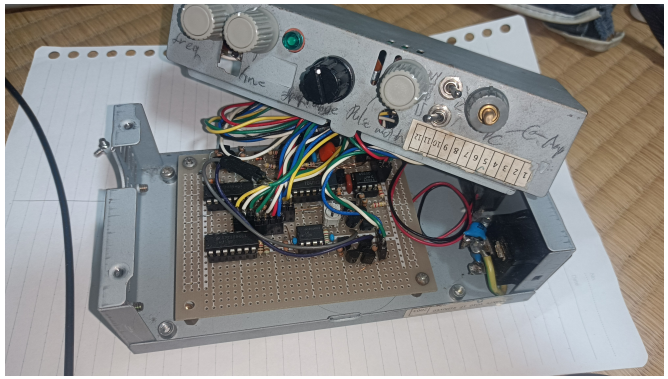
SPICE

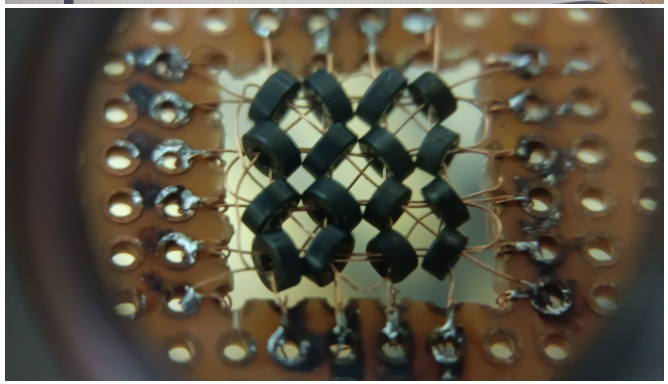
October 31, 2026

自己紹介

なまえ：0x7c00

- 電気系の学部生
- ちょっとしたアナログ回路が趣味
- シェーダーが、VRC 内で唯一アバターに仕込めるスクリプトというだけの理由で、シェーダーをいじっている。





目次

1. SPICE とは
2. 節点解析
3. 修正節点解析
4. 時間積分
5. Newton-Raphson 法
6. 結論

SPICE とは

wikipedia によると...

SPICE

SPICE（スパイス）は、電子回路シミュレータである。カリフォルニア大学バークレー校で 1970 年代前半頃に開発が始まり、以後何回かバージョンアップされた。集積回路の設計に利用可能なように開発されたため...(中略)

- 抵抗、コンデンサ、コイル等受動素子以外にも、トランジスタ、ダイオード、等非線形素子まで取り入れたシミュレーションが可能
- 時間ドメイン・周波数解析・DC 安定点解析・素子定数の最適化 ...できる計算は多岐に渡る
- オープンソース・商用問わず現在も開発されている

SPICE とは

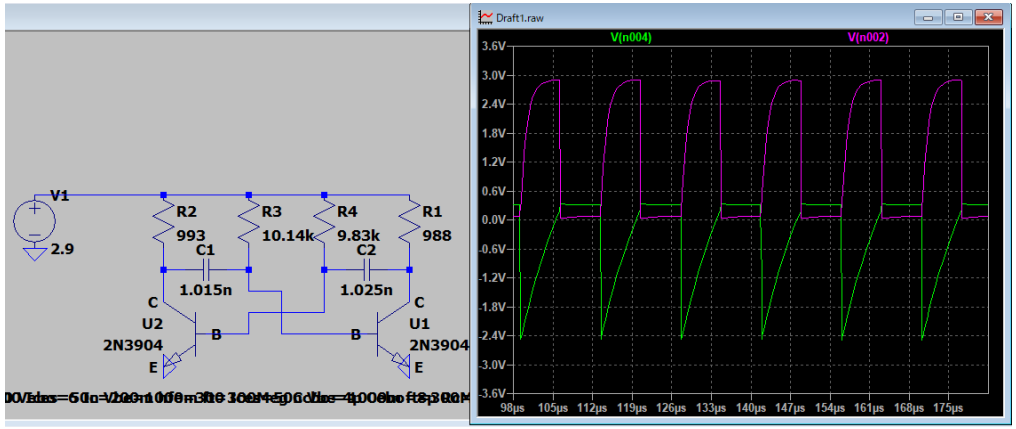


図: LTspice(Analog Devices 社) によるシミュレーション例

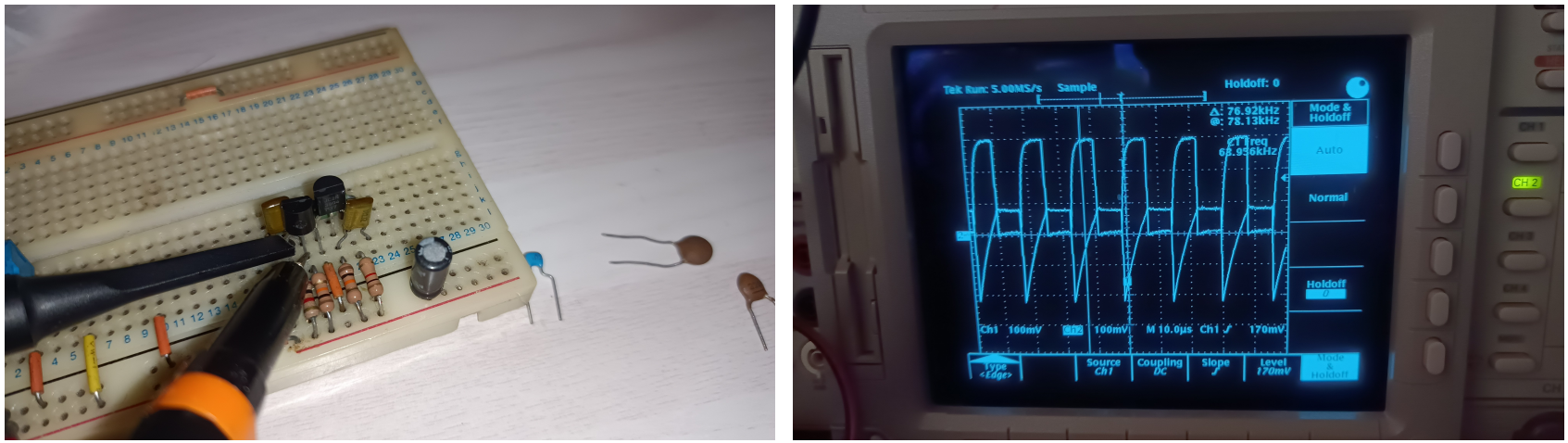


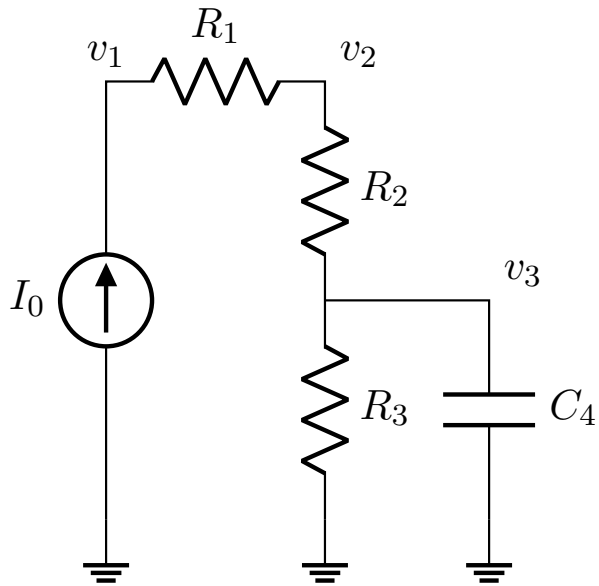
図: 実測例

節点解析

回路の各部の電圧を知りたい
→具体的には n 個の節点に対して、$[v_1, \ldots, v_n]$ を知りたい。
n 個の独立な方程式があれば解けそう。

右の例では、3 つの未知数がある。
キルヒホッフの電流則より、

$$\begin{cases} G_1(v_1 - v_2) = I_0 \\ G_1(v_2 - v_1) + G_2(v_2 - v_3) = 0 \\ G_2(v_3 - v_2) + G_3 v_3 + C_4 \dot{v}_3 \end{cases}$$



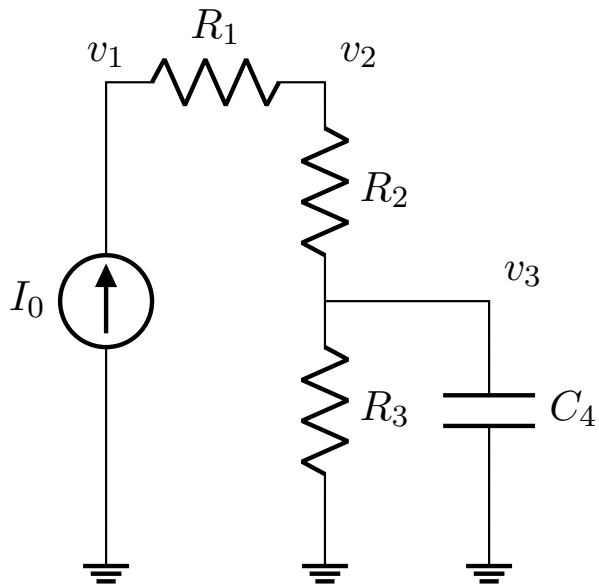
[1] これで独立な式が揃う！

[1] 抵抗値 R ではなく、導電率 G を使っている。$v = Ri$ に対して、$i = Gv$

節点解析

行列で書くと、

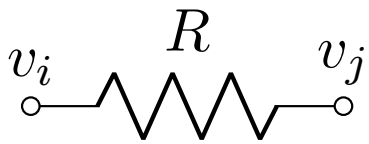
$$\begin{bmatrix} G_1 & -G_1 & 0 \\ -G_1 & G_1+G_2 & -G_2 \\ 0 & -G_2 & G_2+G_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & C \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \end{bmatrix} = \begin{bmatrix} I_0 \\ 0 \\ 0 \end{bmatrix}$$



行列の各行が、各節点に関しての電流則を表している。

節点解析

この行列は、簡単に生成可能



各節点に対して電流則は、各節点から流出する電流の総和を取ると、

$$\text{節点 } v_i \text{に対して：} \ \ldots + G(v_i - v_j) + \ldots = 0$$
$$\text{節点 } v_j \text{に対して：} \ \ldots + G(v_j - v_i) + \ldots = 0$$

行列で書くと

$$\begin{bmatrix} & \downarrow v_i & & \downarrow v_j & \\ & \vdots & & \vdots & \\ \cdots & G & \cdots & -G & \cdots \\ & \vdots & & \vdots & \\ \cdots & -G & \cdots & G & \cdots \\ & \vdots & & \vdots & \end{bmatrix} \begin{bmatrix} \vdots \\ v_i \\ \vdots \\ v_j \\ \vdots \end{bmatrix} = 0$$

節点解析

コンデンサに対しては、

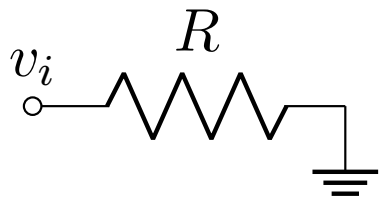
$$\text{節点 } v_i \text{に対して：} \ \ldots + C(\dot{v}_i - \dot{v}_j) + \ldots = 0$$
$$\text{節点 } v_j \text{に対して：} \ \ldots + C(\dot{v}_j - \dot{v}_i) + \ldots = 0$$

行列で書くと

$$\begin{bmatrix} & \downarrow v_i & & \downarrow v_j & \\ & \vdots & & \vdots & \\ \cdots & C & \cdots & -C & \cdots \\ & \vdots & & \vdots & \\ \cdots & -C & \cdots & C & \cdots \\ & \vdots & & \vdots & \end{bmatrix} \begin{bmatrix} \vdots \\ \dot{v}_i \\ \vdots \\ \dot{v}_j \\ \vdots \end{bmatrix} = 0$$

節点解析

接地された素子に対しては、



$$\text{節点 } v_i \text{に対して：} \ \ldots + Gv_i + \ldots = 0$$

行列で書くと、

$$\begin{bmatrix} \cdots & \overset{\downarrow v_i}{\vdots} & \cdots \end{bmatrix} \begin{bmatrix} \vdots \\ v_i \\ \vdots \end{bmatrix} = 0$$

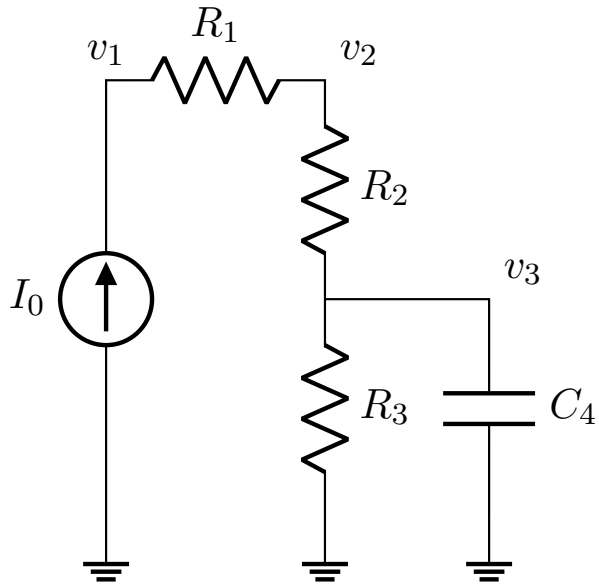
$$\begin{bmatrix} \cdots & G & \cdots \end{bmatrix}$$

コンデンサに対しても同様

節点解析

以前の例では、

$$\begin{bmatrix} G_1 & -G_1 & 0 \\ -G_1 & G_1+G_2 & -G_2 \\ 0 & -G_2 & G_2+G_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & C \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \end{bmatrix} = \begin{bmatrix} I_0 \\ 0 \\ 0 \end{bmatrix}$$



となっている。SPICE において、これはスタンプ (Stamp) と呼ばれている。

修正節点解析

$v, \dot{v}$ のみを含む方程式は簡単に解を計算できる。(そうしたい)
前節までの内容はまさにその典型例。ただ...

- インダクタ (コイル) の回路方程式は、

$$i = \frac{1}{L}\int_0^t v(t)\mathrm{d}t + i(0)$$

  である。

- 定電圧源の回路方程式は、

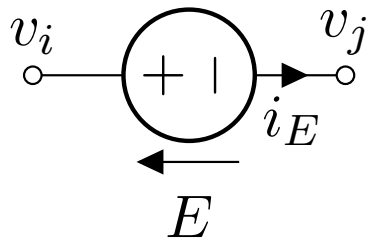
$$v = \text{定数} \quad i = \text{任意}$$

  である。

実は、$v, \dot{v}, i, \dot{i}$ を用いると記述できる。
→修正節点解析 (Modified nodal analysis)

修正節点解析

電圧源に関して、



$$\text{節点 } v_i \text{ の電流則：} \cdots + i_E + \cdots = 0$$
$$\text{節点 } v_j \text{ の電流則：} \cdots + (-i_E) + \cdots = 0$$
$$\text{電圧則：} v_i - v_j = E$$

が成り立つ。

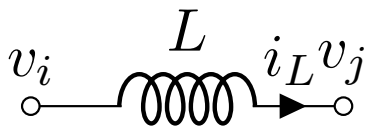
修正節点解析

行列で書くと、

$$
\begin{array}{c}
\begin{array}{ccccc} \quad\downarrow v_i & & \quad\downarrow v_j & & \end{array} \\
\left[\begin{array}{ccccc|c}
 & & & & & \vdots \\
 & \ddots & & & & 1 \\
 & & \ddots & & & \vdots \\
 & & & \ddots & & -1 \\
 & & & & \ddots & \vdots \\
\hline
\cdots & 1 & \cdots & -1 & \cdots & 0
\end{array}\right]
\end{array}
\left[\begin{array}{c} \vdots \\ v_i \\ \vdots \\ v_j \\ \vdots \\ \hline i_E \end{array}\right]
=
\left[\begin{array}{c} \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \hline E \end{array}\right]
$$

修正節点解析

コイルでは



$$\text{節点 } v_i \text{ の電流則}: \cdots + i_L + \cdots = 0$$
$$\text{節点 } v_j \text{ の電流則}: \cdots + (-i_L) + \cdots = 0$$
$$\text{電圧則}: v_i - v_j = L\dot{i_L}$$

が成り立つ。

修正節点解析

行列で書くと、

$$\begin{array}{c} \downarrow v_i \quad \downarrow v_j \\ \left[\begin{array}{cc|c} \multicolumn{2}{c|}{\multirow{2}{*}{$\ddots$}} & \vdots \\ & & 1 \\ & & \vdots \\ & & -1 \\ & & \vdots \\ \hline \cdots & 1 \cdots -1 \cdots & 0 \end{array}\right] \end{array} \left[\begin{array}{c} \vdots \\ v_i \\ \vdots \\ v_j \\ \vdots \\ \hline i_E \end{array}\right] + \begin{array}{c} \downarrow v_i \\ \left[\begin{array}{ccc} \cdots & L & \cdots \end{array}\right] \end{array} \left[\begin{array}{c} \vdots \\ i_E \\ \vdots \end{array}\right] = \left[\begin{array}{c} \vdots \\ \\ \vdots \\ \hline 0 \end{array}\right]$$

修正節点解析

修正接点解析では、様々な素子を記述できる。一般的には

$$A\mathbf{x} + B\dot{\mathbf{x}} + \text{非線形項} - \mathbf{b} = 0$$

という式になる。
$A, B, \mathbf{b}$ などにスタンプを押すことで、回路方程式を機械的に生成可能。
ここから種々の解析を実行できる。

A は普通の回路では正則 (逆行列がある)。
逆に、何もつながっていないノードがあったり、GND から辿れない部品があると、非正則になる。

時間積分

コンピュータで微分方程式の初期値問題を解きたい。
微分を有限区間の差分に置き換えることにより積分。

例として、前進オイラー法

$$\mathbf{x} \to \mathbf{x_n} \quad \dot{\mathbf{x}} \to \frac{\mathbf{x_{n+1}} - \mathbf{x_n}}{\Delta t}$$

後退オイラー法

$$\mathbf{x} \to \mathbf{x_{n+1}} \quad \dot{\mathbf{x}} \to \frac{\mathbf{x_{n+1}} - \mathbf{x_n}}{\Delta t}$$

等色々な差分法がある。

SPICE では、計算上 $\dot{x}$ については解けない (行列 B が正則でない) ことが多いので、後退差分法を用いること都合がよい。

時間積分

差分法の計算精度について考えるため、前進法について考えてみよう。

$$\dot{x} = f(x, t)$$

を解きたい。真の解を

$$x(t_0 + \Delta t) = x(t_0) + \Delta t \dot{x}(t_0) + \frac{\Delta t^2}{2} \ddot{x}(t_0) + \cdots$$

と表す。これを、既知の $x(t_0), f\left(x(t_0), t_0\right)$ や、それ以前の計算結果も用いて近似したい。

時間積分

$x(t_0 + \Delta t) \simeq x(t_0) + \Delta t f\left(x(t_0), t_0\right)$ と近似してみる。誤差は、

$$x(t_0 + \Delta t) - \{x(t_0) + \Delta f\left(x(t_0), t_0\right)\} = \frac{\Delta t^2}{2}\ddot{x}(t_0) + \cdots$$

実際の数値計算は、Δt を十分小さくすることになるので、まとめて $\mathcal{O}(\Delta t^2)$ と書かれたり、1 次精度と言われたりする。

より以前の計算結果まで用いると、より高次の項まで消すことができる。

Table: ギア法の積分公式 $(x = x_{n+1})$[1]

次数	積分公式
1	$\Delta t\dot{x} = x_{n+1} - x_n$
2	$\Delta t\dot{x} = \frac{3}{2}x_{n+1} - 2x_n + \frac{1}{2}x_{n-1}$
3	$\Delta t\dot{x} = \frac{11}{6}x_{n+1} - 3x_n + \frac{3}{2}x_{n-1} - \frac{1}{3}x_{n-2}$
$\vdots$	$\vdots$

時間積分

電子回路では、急に変化率が大きくなったりする。ただ平坦なところで無駄に細かく時間を刻むのも無駄
時間ステップを「空気を読んで」調整する必要。

誤差 E を推定して、$E = \mathcal{O}(\Delta t^{n+1}) < (\text{定数})$ となるよう調整

誤差推定には

- Δt で 1 回時間発展したものと、$\Delta t/2$ で 2 回時間発展したものの差を誤差とする方法
- 今までの結果の n 次の多項式近似 (予測子) と、n 次の時間発展 (修正子) の差を誤差とする方法

等がある。

※正直この辺は「非線形問題を解くためのその場しのぎ」ぐらいに思っておいてください。色々工夫しても適切に計算を回すのはかなり難しいです。

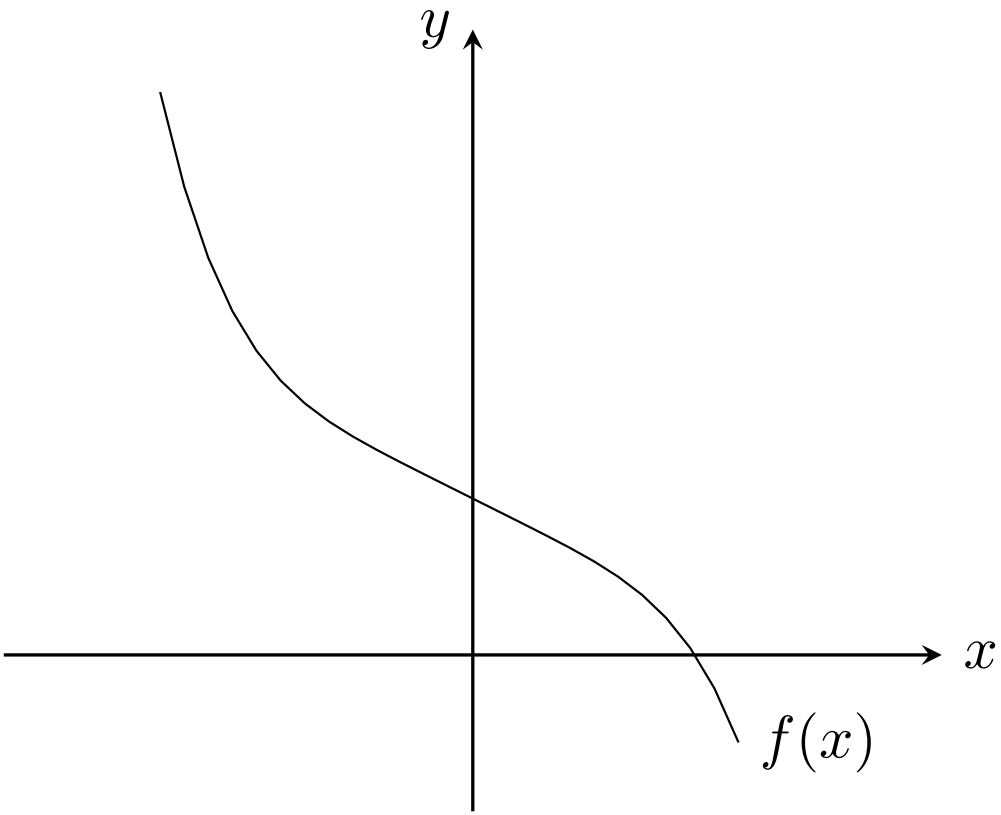
Newton-Raphson 法

ここまでで、シミュレーションは、n 変数 n 次連立非線形方程式
$\mathbf{F}(\mathbf{x}_{n+1}) = 0$ を解く問題に帰着した。

こうした問題を解くのに、Newton-Raphson 法 (NR 法) が使われている。

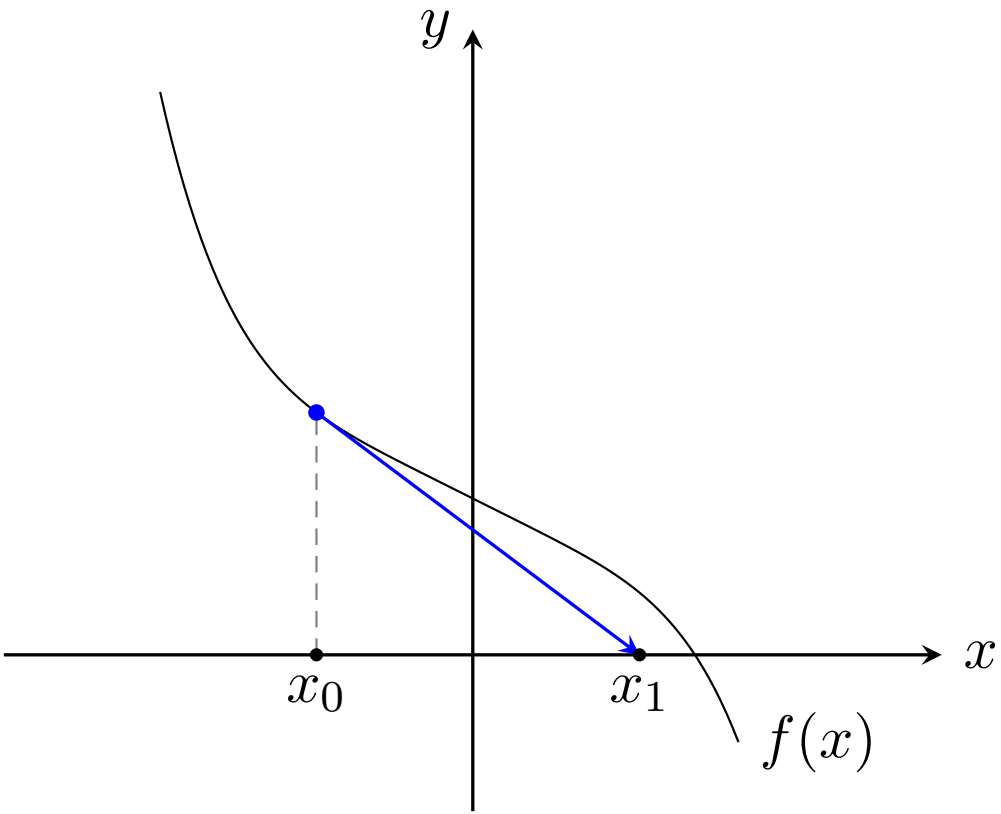
Newton-Raphson 法

例として、1 変数の $f(x) = 0$ の解を探索することを考える。



Newton-Raphson 法

適当に開始点を取り、一次近似する。

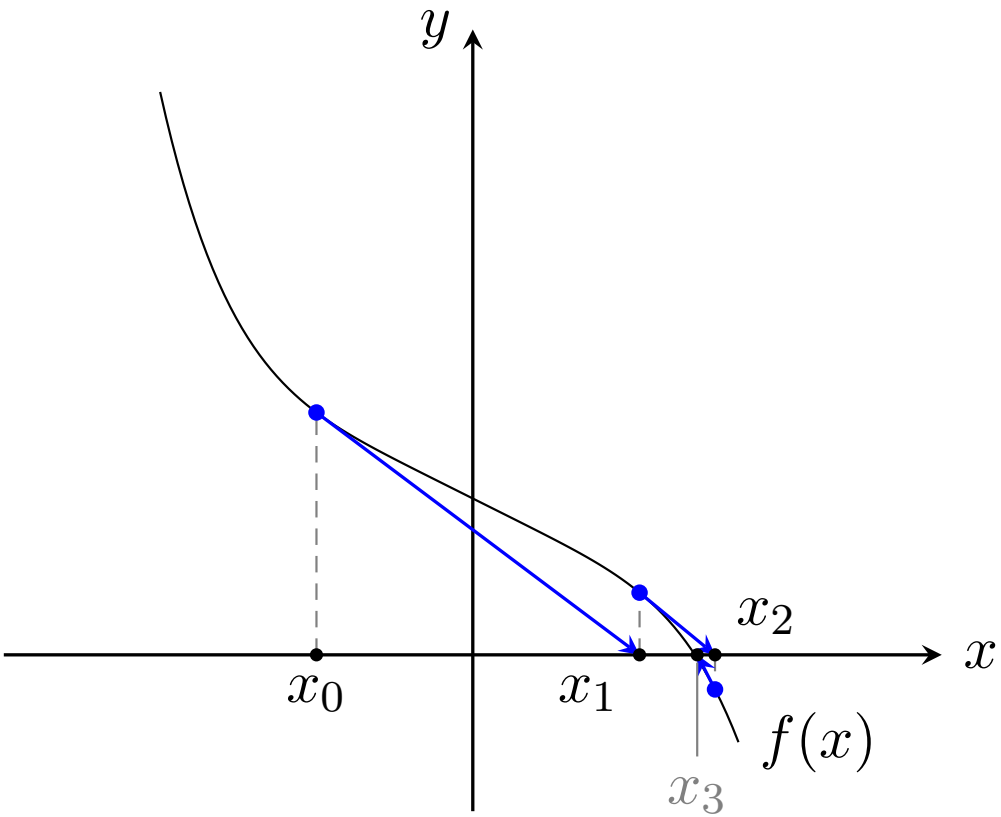


$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

である。$f'(x)$ はわかっている必要あり。

Newton-Raphson 法

これを繰り返す。



Newton-Raphson 法

多変数関数では、ヤコビアンの逆行列を使って、

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left(\frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} \right)^{-1}_{\mathbf{x}=\mathbf{x}_n} \mathbf{F}(\mathbf{x}_n)$$

とすればよい。

NR 法は、よい初期値で開始すればかなりすぐに収束する。
逆に初期値によっては収束しないこともある。
また、解が複数ある場合、どの解に収束するのかは全くわからない。

結論

結局 SPICE は、

修正接点解析で回路方程式を生成
↓
微分方程式を時間離散化する
↓
NR 法を回して解く
↓
誤差評価、時間ステップ調整

というサイクルで、計算を回している。

計算が収束しないこともあるので、色々なソルバーにより様々な工夫がなされている。

参考文献 I

[1] 田中 衞牛田 明夫.
電子回路シミュレーション.
コロナ社, 2002.