ModelNgspicer User Guide

Version 1.0

Developed by ペE

1　はじめに

ModelNgspicer は、ngspice をベースにした GUI アプリケーションであり、回路設計の最適化やデバイスモデリングにおける作業効率の向上を目的としています。GUI は Python の Qt ライブラリ（PySide6）を用いて構築されており、マウスホイールや数値入力によってモデルパラメータを調整し、特性の変化をリアルタイムでグラフ表示することが可能です。

Ngspice は、オープンソースの SPICE 系回路シミュレータであり、DC 解析、AC 解析、過渡解析、ノイズ解析、S パラメータ解析など、多彩な解析機能を備えています。また、Verilog-A や BSIM などの業界標準モデルにも対応しており、高精度なデバイスモデリングが可能です。コマンドラインベースで動作するため、スクリプトによる複雑な計算処理や自動解析にも適しています。

Ngspice の公式サイト：
https://ngspice.sourceforge.io

2　インストール

2.1　動作環境

ModelNgspicer は、以下の環境で動作確認されています。

- OS：Windows 10 以降（64 ビット）
- Python：バージョン 3.13 以降
- 必要なライブラリ：PySide6、PyQtGraph、numpy
- Ngspice：バージョン 43 以降を推奨

2.2　インストール手順（Windows）

Windows 環境でのインストール手順を以下に示します。ここでは、Windows 用パッケージマネージャ「Chocolatey」を用いたインストールを紹介しますが、公式サイトからダウンロードしてインストールしても OK です。

1. Chocolatey のインストール
管理者権限で PowerShell を開き、以下のコマンドを実行します。詳細は Chocolatey の公式ページ（https://chocolatey.org/install）を参照してください。

```Powershell
Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/
install.ps1'))
```

2. Python のインストール

Chocolatey を用いて Python をインストールします。管理者権限の PowerShell またはコマンドプロンプトで以下のコマンドを実行します。インストール後、**python** および **pip** コマンドが使用可能であることを確認してください。

```Powershell
choco install python
```

3. ngspice のインストール

Chocolatey を用いて ngspice をインストールします。管理者権限の PowerShell またはコマンドプロンプトで以下のコマンドを実行します。インストール後、**ngspice** および **ngspice_con** コマンドが使用可能であることを確認してください。

```Powershell
choco install ngspice
```

4. 必要ライブラリのインストール

以下のコマンドを実行して、必要な Python ライブラリをインストールします。

```Powershell
pip install pyside6 pyqtgraph numpy
```

以上で、セットアップは完了です。プロジェクトフォルダ内の **run.vbs** を実行することで、ModelNgspicer を起動できます。

3　GUI と基本操作

3.1　画面構成の概要

以下の図は、ModelNgspicer の画面構成を示しています。

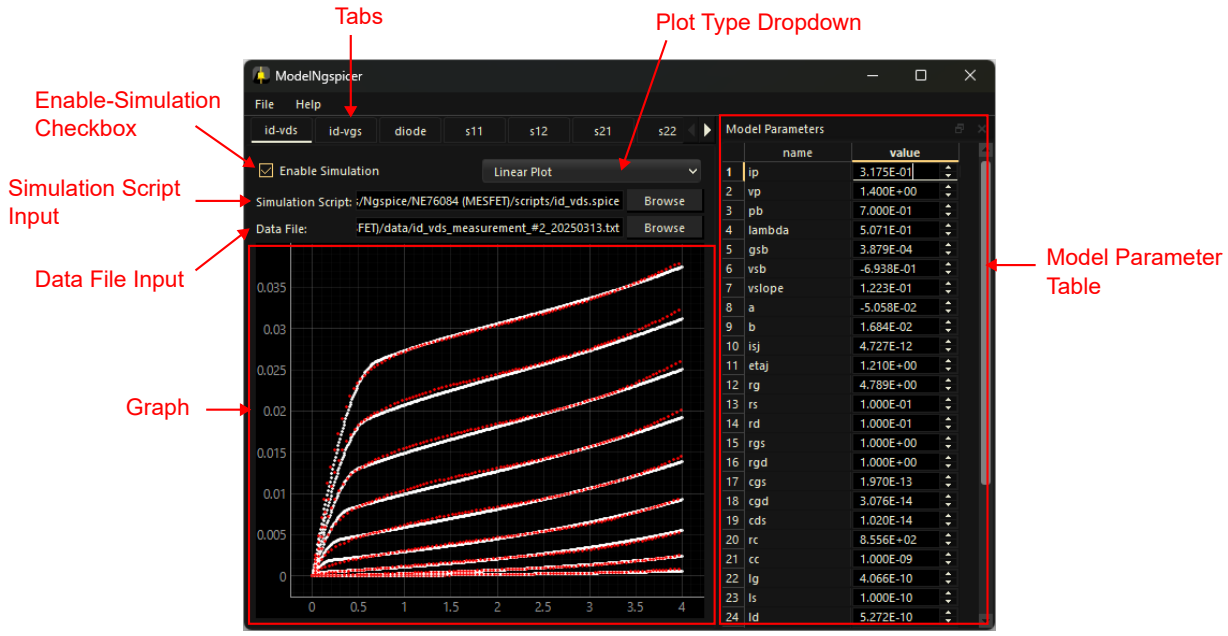

Fig. 3-1　GUI 画面の構成

- **タブパネル（Tabs）**
  画面上部には 10 個のタブが並んでおり、それぞれ異なるシミュレーション設定を保持できます。タブ名はダブルクリックで変更可能です。右クリックから「Pop out」を選択すると、タブの内容を別ウィンドウとして表示でき、閉じると元のタブ位置に戻ります。複数のウィンドウを同時に開いて平行してパラメータ調整を行うことも可能です。
- **モデルパラメータテーブル（Model Parameter Table）**
  モデルパラメータの一覧が表示されます。メニューバーの「File > Load Parameters...」からパラメータ定義ファイルを読み込むことで、テーブルに反映されます。各パラメータの値はスピンボックスに表示され、カーソルを合わせてマウスホイールを操作することで値を変更できます。Ctrl キーを押しながらマウスホイール操作をすると、粗い調整が可能です。
- **シミュレーション有効化チェックボックス（Enable Simulation Checkbox）**
  現在表示されているタブのシミュレーションを有効化／無効化できます。有効化されている場合、パラメータを変更するたびにシミュレーションが実行されます。不要なタブのシミュレーションを無効化しておくことで、実行時間を短縮できる可能性があります。
- **シミュレーションスクリプト入力（Simulation Script Input）**
  ngspice のスクリプトファイルを選択するフィールドです。「Browse」ボタンを押すとファイルダイアログが開き、スクリプト（**.spice**、**.sp**、**.cir**）を選択できます。選択されたスクリプトは Python から **ngspice_con** コマンドを用いて実行され、同名の**.txt** ファイルとして結果が出力されます。この結果は自動的にグラフ領域にプロットされます。

- **データファイル入力（Data File Input）**
  シミュレーション結果との比較用に、外部データファイルを読み込むことができます。読み込まれたデータは、シミュレーション結果と同じグラフ上にプロットされます。測定値やデータシートの読み値、目標値などを重ねて表示したい時に便利です。
- **プロット形式選択（Plot Type Dropdown）**
  グラフの表示形式を選択します。Linear Plot、Log-Log、Semi-Log X、Semi-Log Y、Smith Chart の中から選択できます。
- **グラフ領域（Graph）**
  シミュレーション結果は白点、データファイルの値は赤点でプロットされます。

3.2 基本操作フロー

ModelNgspicer を使用するにあたり、ユーザーは以下の最低 2 つのファイルを用意する必要があります。これらのファイル形式については、次章にて説明します。

- パラメータ定義ファイル（**.txt**）
- ngspice スクリプト（**.spice**、**.sp**、**.cir**）

ModelNgspicer の基本的な使用手順は以下の通りです。

1. モデルパラメータの読み込み
   メニューバーの「File > Load Parameters...」から、パラメータ定義ファイル（**.txt**）を選択します。モデルパラメータが読み込まれ、画面右のモデルパラメータテーブルに表示されます。
2. ngspice スクリプトの選択
   シミュレーションスクリプト入力の「Browse」ボタンから、使用する ngspice スクリプト（**.spice**、**.sp**、**.cir**）を選択します。
3. シミュレーションの有効化
   シミュレーション有効化チェックボックスをオンにします。
4. モデルパラメータの調整
   モデルパラメータテーブルに表示されたスピンボックスを操作することで値を変更します。
   - マウスホイール操作により微調整
   - Ctrl キーを押しながらマウスホイール操作により粗調整
5. 比較データの読み込み（任意）
   データファイル入力の「Browse」ボタンから、外部データファイル（測定結果や目標値など）を指定します。
6. プロット形式の選択
   プロット形式選択のドロップダウンリストから、グラフの表示形式を選択します。
7. 結果の確認
   グラフ領域にて、プロットされたシミュレーション結果（白点）と外部データ（赤点）を確認します。
8. モデルパラメータの保存
   メニューバーの「File > Save Parameters...」から、調整後のモデルパラメータを保存します。

4　ファイル形式

4.1　パラメータ定義ファイル

モデルパラメータは、プレーンテキストファイル（**.txt**）を用いて定義します。ファイル内の各行は、以下のように先頭に**+**記号を付けて記述します。

```
Plain Text
+ is = 1e-14
+ n = 1.5
+ rs = 0.5
+ …
```

この**+**記号は SPICE 構文における行継続記号であり、前の行からの継続を意味します。この形式を採用することで、**.param** や**.model** ステートメントにそのまま展開可能となり、ngspice スクリプトへの組み込みが容易になります。

例 1　**.param** 文に展開する場合：

```
Ngspice Script
.param
.include model.txt
```

上のスクリプトのように**.param** の直下に**.include** 文を記述すると、ngspice は内部的に以下のように展開します。したがって、パラメータファイル **model.txt** の内容が**.param** の引数として認識され、ngspice スクリプトに簡潔に埋め込むことができます。

```
Ngspice Script
.param
+ is = 1e-14
+ n = 1.5
+ rs = 0.5
+ …
```

例 2　**.model** 文に展開する場合：

```
Ngspice Script
.model DIODE1 D (
.include model.txt
+ )
```

上の例では、パラメータファイル **model.txt** の内容が **DIODE1** のモデルパラメータとして適用されます。このようにして、カスタムデバイスモデルを作成することが可能となります。

4.2 シミュレーション結果とデータファイル

ModelNgspicer では、シミュレーション結果と外部データファイルの両方に、共通のプレーンテキスト形式（**.txt**）を使用します。ファイル構造は以下となります。

- ヘッダー行なし
- 1 列目：X 軸データ
- 2 列目以降：Y 軸データ
- 区切り文字：スペースまたはタブ

例：

```
Plain Text
1e6  -3.2  -1.1
2e6  -3.5  -1.3
3e6  -3.8  -1.6
…
```

シミュレーション結果の出力方法：
ngspice スクリプト内で **wrdata** コマンドを使用することで、シミュレーション結果を出力できます。出力ファイル名は、スクリプトファイルと同じ名前＋**.txt** 拡張子にする必要があります。この命名規則により、ModelNgspicer が自動的に結果ファイルを認識し、グラフ領域にプロットします。

出力例（ngspice スクリプト）：

```
Ngspice Script
set wr_singlescale
wrdata iv.txt i(v1) i(v2)
```

この例では、**iv.spice** というスクリプトを実行すると、同名の **iv.txt** という結果ファイルが生成されます。出力結果には、1 列目にスケールベクトル、2〜3 列目に電流値 **i(v1)**と **i(v2)**が記録されます。

注意点：

- ファイル名の一致は必須です。スクリプト名と結果ファイル名が一致していない場合、自動プロットが行われません。
- データの整合性が必要です。不正な形式や不整合のあるデータが含まれていると、グラフが正しく描画されない可能性があります。

5　活用例

5.1　LC バンドパスフィルタの設計

この節では、ModelNgspicer を用いた LC バンドパスフィルタの設計例を紹介します。

Fig. 5-1 に回路図を示します。ノード名を **n01**～**n05** とラベル付けしており、モデルパラメータ **Cval1**、**Lval2**、**Cval3**、**Lval4** を使用します。今回は、バンドパスフィルタの中心周波数が 100 MHz となるように設計を進めてみましょう。

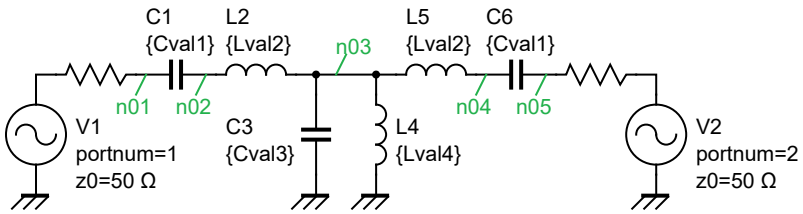

Fig. 5-1　LC バンドパスフィルタの回路図

以下のように、パラメータ定義ファイル（**model_initial.txt**）と、ngspice スクリプト（**bpf1.spice**）を作成します。

ファイル名：**model_initial.txt**

```
+ Cval1=10p
+ Lval2=300n
+ Cval3=253p
+ Lval4=10n
```

ファイル名：**bpf1.spice**

```
 1 bpf1.spice
 2
 3 .param
 4 .include model.txt
 5
 6 V1  n01 0       dc 0 portnum 1 z0 50
 7 V2  n05 0       dc 0 portnum 2 z0 50
 8 C1  n01 n02     {Cval1}
 9 L2  n02 n03     {Lval2}
10 C3  n03 0       {Cval3}
11 L4  n03 0       {Lval4}
12 L5  n03 n04     {Lval2}
13 C6  n04 n05     {Cval1}
14
15 .control
16 sp lin 200 50Meg 150Meg
17
18 set wr_singlescale
```

```
19 wrdata bpf1.txt db(s_2_1) db(s_1_1)
20
21 .endc
22 .end
```

スクリプトの解説：
1 行目：ngspice では 1 行目はコメントとして扱われます（今回はファイル名を記載）。
3 行目：**.param** 文を開始します。
4 行目：**.include** 文により、**.param** の引数としてモデルパラメータを展開します。
6〜13 行目：回路構成をネットリストで記述しています。電圧源 **V1** と **V2** は、**portnum** を指定することで RF ポートとして扱われ、**z0** によって基準インピーダンス（ここでは 50 Ω）を設定しています。
15 行目：**.control** により制御セクション（バッチ処理）の開始を示します。
16 行目：**sp lin 200 50Meg 150Meg** は S パラメータ解析コマンドで、50 MHz〜150 MHz の範囲を 200 ポイントで線形スイープします。
18 行目：**wrdata** の出力形式を 1 列目=X 軸データ、2 列目以降=Y 軸データとします。
19 行目：S21 および S11 の結果を **bpf1.txt** に出力します。出力ファイルはスクリプトファイルと同名にする必要があります。
21 行目：**.endc** は制御セクションの終了を示します。
22 行目：**.end** はスクリプト全体の終了を示します。

以上で ModelNgspicer を使用する準備が整ったので、実際に動かしてみましょう。メニューバーの「File > Load Parameters...」から **model_initial.txt** を読み込みます。次に、シミュレーションスクリプト入力の「Browse」ボタンから **bpf1.spice** を選択します。

読み込み後の画面は Fig. 5-2 のようになり、グラフ領域には S11 と S21 の周波数特性が表示されています。この時点では、モデルパラメータは初期値のままであるため、フィルタ特性がややいびつな形になっていることが分かります。そこで、**Cval1** の値を調整することで、フィルタの中心周波数を 100 MHz に近づけてみましょう。

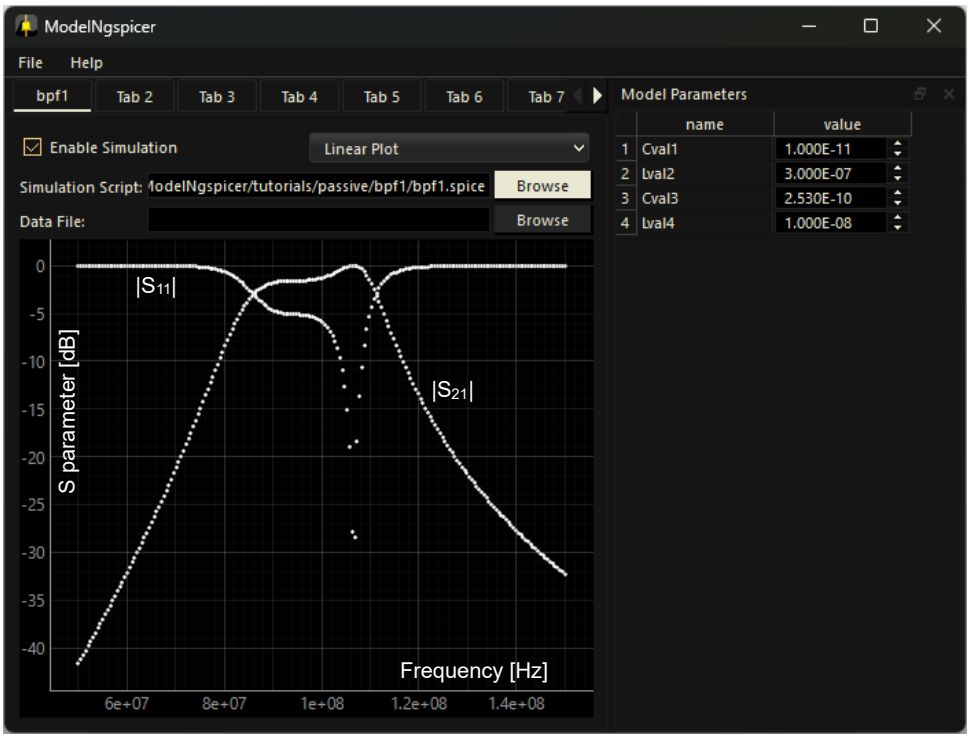

Fig. 5-2　LC バンドパスフィルタの設計（調整前）

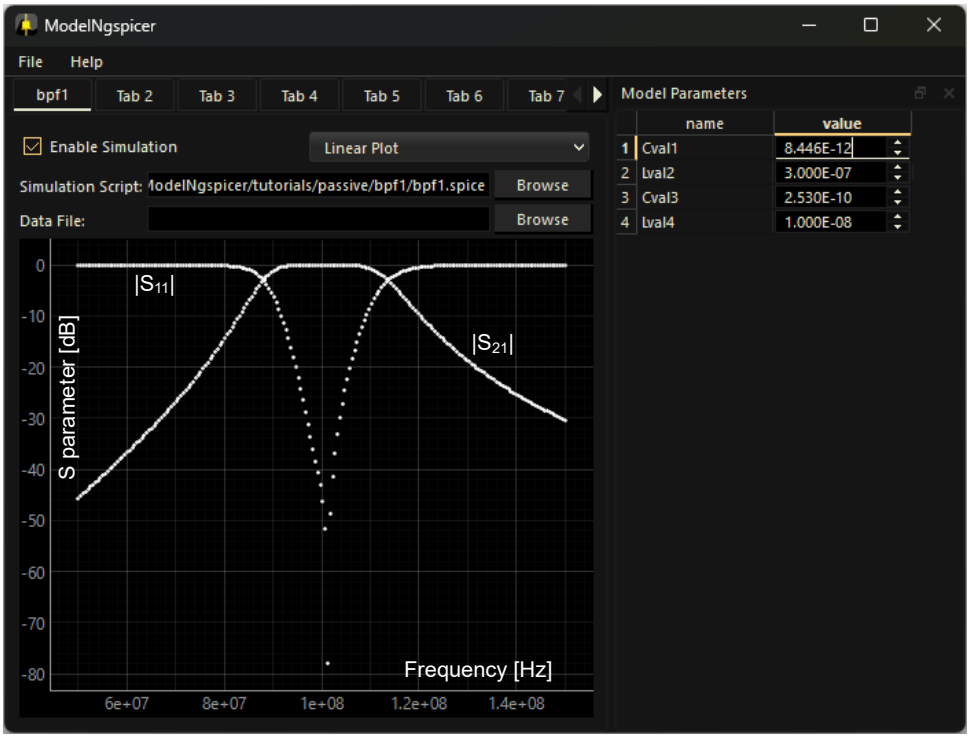

Fig. 5-3　LC バンドパスフィルタの設計（調整後）

調整後の画面を Fig. 5-3 に示します。通過帯域内の特性が滑らかになり、中心周波数付近で最大ゲインが得られていることが確認できます。

最後に、調整したモデルパラメータを保存しておきましょう。メニューバーの「File > Save Parameters...」から、現在のパラメータ値をテキストファイルとして保存できます。

5.2 ダイオードのモデリング

この節では、ModelNgspicer を使用したダイオードのモデリング例を紹介します。実際に、Onsemi 製 1N4148 のデータシートを基にモデルパラメータを合わせていきます。ここでは、以下の特性に着目してモデリングを行います。

- 順方向・逆方向電流–電圧特性（I_F–V_F, I_R–V_R）
- 接合容量–逆電圧特性（C_T–V_R）
- 逆回復時間（t_{rr}）

Onsemi, 1N4148 データシート：
https://www.onsemi.com/products/discrete-power-modules/small-signal-switching-diodes/1n4148

5.2.1 モデルパラメータ

以下に、調整対象とする SPICE モデルのパラメータ一覧およびパラメータファイル（**model_final.txt**）を示します。

Junction DC parameters

Name	Parameter	Units	Default	Final value
IS	Saturation current	A	1.0e-14	4.277E-09
JSW	Sidewall saturation current	A	0.0	0.000E+00
N	Emission coefficient	-	1.0	1.912E+00
RS	Ohmic resistance	Ω	0.0	7.416E-01
BV	Reverse breakdown voltage	V	infinity	1.42E+02
IBV	Current at breakdown voltage	A	1.0e-3	1.000E-03
NBV	Breakdown Emission Coefficient	-	N	4.784E+01
IKF	Forward knee current	A	0.0	0.000E+00
IKR	Reverse knee current	A	0.0	5.000E-07
JTUN	Tunneling saturation current	A	0.0	3.987E-11
JTUNSW	Tunneling sidewall saturation current	A	0.0	0.000E+00
NTUN	Tunneling emission coefficient	-	30	4.914E+02
XTITUN	Tunneling saturation current exponential	-	3	3.000E+00
KEG	EG correction factor for tunneling	-	1.0	1.000E+00
ISR	Recombination saturation current	A	1.0e-14	1.000E-14
NR	Recombination current emission coefficient	-	2.0	2.000E+00

Junction capacitance parameters

Name	Parameter	Units	Default	Final value
CJO	Zero-bias junction bottom-wall capacitance	F	0.0	8.695E-13
CJP	Zero-bias junction sidewall capacitance	F	0.0	0.000E+00

FC	Coefficient for forward-bias depletion bottom-wall capacitance formula	-	0.5	5.000E-01
FCS	Coefficient for forward-bias depletion sidewall capacitance formula	-	0.5	5.000E-01
M	Area junction grading coefficient	-	0.5	2.266E-02
MJSW	Periphery junction grading coefficient	-	0.33	3.300E-01
VJ	Junction potential	V	1.0	7.000E-01
PHP	Periphery junction potential	V	1.0	1.000E+00
TT	Transit-time	sec	0.0	4.121E-09

Temperature effects

Name	Parameter	Units	Default	Final value
EG	Activation energy	eV	1.11 (Si) 0.69 (SBD) 0.67 (Ge)	1.110E+00
XTI	Saturation current temperature exponent	-	3.0 (pn) 2.0 (SBD)	3.000E+00
TNOM	Parameter measurement temperature	°C	27	2.700E+01

Noise parameters

Name	Parameter	Units	Default	Final value
KF	Flicker noise coefficient	-	0	0.000E+00
AF	Flicker noise exponent	-	1	1.000E+00

Scaling factors

Name	Parameter	Units	Default	Final value
area	Scaling factor for area	-	1	1.000E+00
pj	Scaling factor for perimeter	-	1	1.000E+00

ファイル名：**model_final.txt**

```
+ is=4.277E-09
+ jsw=0.000E+00
+ n=1.912E+00
+ rs=7.416E-01
+ bv=1.270E+02
+ ibv=0.000E+00
+ nbv=4.784E+01
+ ikf=0.000E+00
+ ikr=5.000E-07
+ jtun=3.987E-11
+ jtunsw=0.000E+00
+ ntun=4.914E+02
+ xtitun=3.000E+00
+ keg=1.000E+00
+ isr=1.000E-14
```

```
+ nr=2.000E+00
+ cjo=8.695E-13
+ cjp=0.000E+00
+ fc=5.000E-01
+ fcs=5.000E-01
+ m=2.266E-02
+ mjsw=3.300E-01
+ vj=7.000E-01
+ php=1.000E+00
+ tt=4.121E-09
+ eg=1.110E+00
+ xti=3.000E+00
+ tnom=2.700E+01
+ kf=0.000E+00
+ af=1.000E+00
+ area=1.000E+00
+ pj=1.000E+00
```

5.2.2 順方向電流–電圧特性

まずは、順方向電流–電圧特性（I_F–V_F）を基にモデルパラメータを合わせていきます。以下に、シミュレーションに使用する測定回路（Fig. 5-4）と ngspice スクリプト（**if-vf.spice**）を示します。

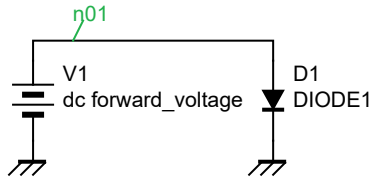

Fig. 5-4　順方向電流–電圧特性（I_F–V_F）の測定回路

ファイル名：**if-vf.spice**

```
 1 if-vf.spice
 2
 3 .model DIODE1 D (
 4 .include model.txt
 5 + )
 6
 7 V1  n01 0   dc 0
 8 D1  n01 0   DIODE1
 9
10 .control
11 option TEMP=25
12 dc V1 0.3 1.5 0.01
13
14 set wr_singlescale
15 wrdata if-vf.txt -i(V1)
16 .endc
17 .end
```

スクリプトの解説：
1 行目：ngspice では 1 行目はコメントとして扱われます（今回はファイル名を記載）。
3〜5 行目：**.model** 文を使用してダイオードモデル **DIODE1** 文を定義しています。**.include** 文によりモデルパラメータを引数として展開します。
7〜8 行目：測定回路をネットリストで記述しています。
10 行目：**.control** により制御セクション（バッチ処理）の開始を示します。
11 行目：**option** により周囲温度 **TEMP** を 25°C に設定します。
12 行目：**dc V1 0.3 1.5 0.01** は DC 解析コマンドで、電圧源 **V1** に対して 0.3 V〜1.5 V の範囲を 0.01 V 刻みで線形スイープします。
14 行目：**wrdata** の出力形式を 1 列目=X 軸データ、2 列目以降=Y 軸データとします。
15 行目：結果を **if-vf.txt** に出力します。
16 行目：**.endc** は制御セクションの終了を示します。
17 行目：**.end** はスクリプト全体の終了を示します。

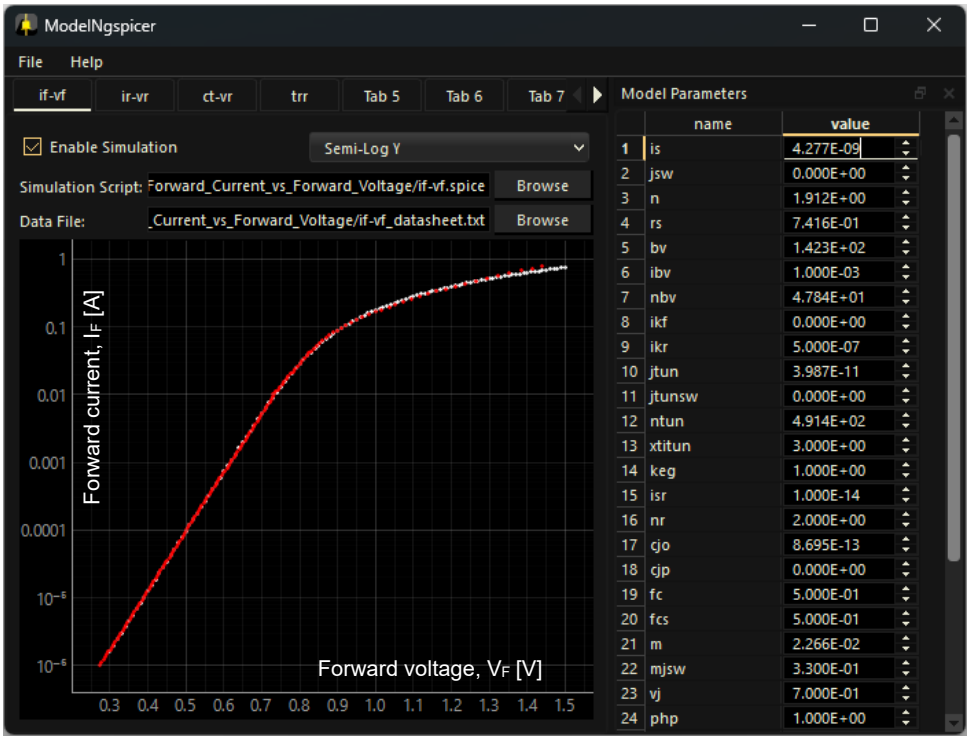

Fig. 5-5　順方向電流–電圧特性のパラメータ調整
（赤点：データシートの読み値、白点：シミュレーション結果）

モデルパラメータの調整：
データシートの順方向電流–電圧特性に合うように、モデルパラメータ **IS**（saturation current）、**N**（emission coefficient）、および **RS**（ohmic resistance）を調整します。低電流領域において、**IS** と **N** はそれぞれグラフの切片と傾きに影響します。大電流領域においては、**RS** が支配的になります。

調整後の順方向電流–電圧特性を Fig. 5-5 に示します。グラフの赤点はデータシートの読み値、白点はシミュレーション結果を示しており、両者の特性はよく一致しています。

5.2.3 逆方向電流–電圧特性

続いて、逆方向電流–電圧特性（I_R–V_R）を基にモデルパラメータを合わせていきます。以下に、使用する測定回路（Fig. 5-6）と ngspice スクリプト（**ir-vr.spice**）を示します。

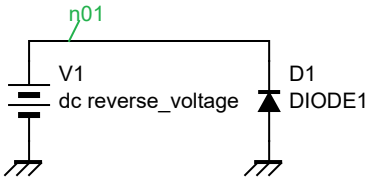

Fig. 5-6　逆方向電流–電圧特性（I_R–V_R）の測定回路

ファイル名：**ir-vr.spice**

```
1  ir-vr.spice
2
3  .model DIODE1 D (
4  .include model.txt
5  + )
6
7  V1  n01 0   dc 0
8  D1  0   n01 DIODE1
9
10 .control
11 option TEMP=25 GMIN=4e-10
12 dc V1 10 146 1
13
14 set wr_singlescale
15 wrdata ir-vr.txt -i(V1)
16 .endc
17 .end
```

スクリプトの解説：
1 行目：ngspice では 1 行目はコメントとして扱われます（今回はファイル名を記載）。
3～5 行目：**.model** 文を使用してダイオードモデル **DIODE1** 文を定義しています。**.include** 文によりモデルパラメータを引数として展開します。
7～8 行目：測定回路をネットリストで記述しています。
10 行目：**.control** により制御セクション（バッチ処理）の開始を示します。
11 行目：**option** により周囲温度 **TEMP** を 25°C、最小コンダクタンス **GMIN** を 4e-10 Ω^{-1} に設定しています。**GMIN** はダイオードのリーク電流を再現しており、データシートの特性に合うよう値を調整しています。
12 行目：**dc V1 10 146 1** は DC 解析コマンドで、電圧源 **V1** に対して 10 V～146 V の範囲を 1 V 刻みで線形スイープします。
14 行目：**wrdata** の出力形式を 1 列目=X 軸データ、2 列目以降=Y 軸データとします。
15 行目：結果を **ir-vr.txt** に出力します。
16 行目：**.endc** は制御セクションの終了を示します。
17 行目：**.end** はスクリプト全体の終了を示します。

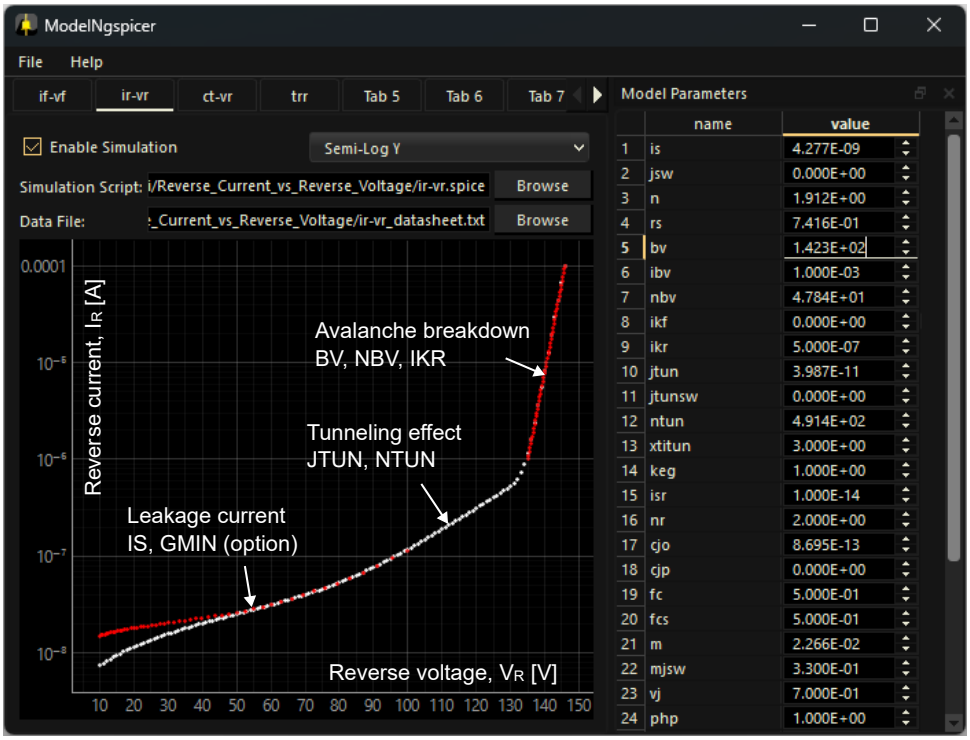

Fig. 5-7　逆方向電流–電圧特性のパラメータ調整
（赤点：データシートの読み値、白点：シミュレーション結果）

モデルパラメータの調整：
ここでは、モデルパラメータ **JTUN**（tunneling saturation current）、**NTUN**（tunneling emission coefficient）、**BV**（reverse breakdown voltage）、**NBV**（breakdown emission coefficient）、**IKR**（reverse knee current）および環境変数 **GMIN**（minimum conductance）を調整します。

逆方向電流–電圧特性を再現するためには、各電圧領域における支配的な物理現象を意識しながら、パラメータを調整すると効率的です。

- 低電圧領域（～70 V）
  この領域のリーク電流は、主に **IS** と **GMIN** によって決まります。しかし、**IS** は既に順方向電流–電圧特性で調整済みであるため、**GMIN** のみ調整することとします。
- 中電圧領域（70～130 V）
  この領域では、電流の傾きが増加する傾向が見られます。この挙動を再現するため、トンネル電流に関係する **JTUN** と **NTUN** を調整します。
- 高電圧領域（130 V 以上）
  この領域では、アバランシェ降伏による急激な電流増加が支配的となります。降伏電圧 **BV** と電流の傾きを決める **NBV** を調整します。**IKR** はアバランシェ降伏の立ち上がり付近（~0.5 µA）に合わせます。

調整後の逆方向電流–電圧特性を Fig. 5-7 に示します。グラフの赤点はデータシートの読み値、白点はシミュレーション結果を示しており、両者の特性はよく一致しています。

5.2.4 接合容量–逆電圧特性

次に、接合容量–逆電圧特性（C_T–V_R）に基にモデルパラメータを合わせていきます。以下に、使用する測定回路（Fig. 5-8）とシミュレーションスクリプト（**ct-vr.spice**）を示します。AC 解析における電流値をピックアップするための直列抵抗 R1 を挿入しています。

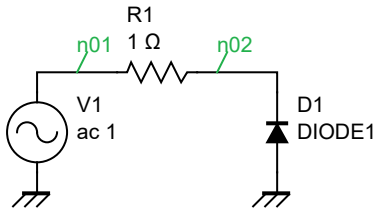

Fig. 5-8　接合容量–逆電圧特性（C_T–V_R）の測定回路

ファイル名：**ct-vr.spice**

```
ct-vr.spice

.model DIODE1 D (
.include model.txt
+ )

V1  n01 0   dc 0 ac 1
R1  n01 n02 1
D1  0   n02 DIODE1

.control
* Reverse voltage sweep settings
let st = 0
let sp = 15
let step = 0.1

let loop_index = 0
let loop_count = (sp-st)/step
let capacitance = vector(loop_count)
let reverse_voltage = st+step*vector(loop_count)

while loop_index lt loop_count
    alter V1 dc reverse_voltage[loop_index]
    ac lin 1 1Meg 1Meg

    $ Calculate capacitance from admittance
    let Y11 = v(n01,n02)/v(n02)
    let capacitance[loop_index] = abs(imag(Y11)/(2*pi*frequency))/1e-12
    let loop_index = loop_index + 1
end

setscale capacitance reverse_voltage
wrdata ct-vr.txt capacitance
.endc
.end
```

スクリプトの解説：
1 行目：ngspice では 1 行目はコメントとして扱われます（今回はファイル名を記載）。
3〜5 行目：**.model** 文を使用してダイオードモデル **DIODE1** 文を定義しています。**.include** 文によりモデルパラメータを引数として展開します。
7〜9 行目：測定回路をネットリストで記述しています。
11 行目：**.control** により制御セクション（バッチ処理）の開始を示します。
13〜15 行目：電圧スイープの設定：**st**（開始）、**sp**（終了）、**step**（間隔）を定義しています。
17〜18 行目：ループ処理で使用する変数 **loop_index**、**loop_count** を定義しています。
19〜20 行目：出力結果を保持するベクトル変数 **capacitance** および **reverse_voltage** を定義しています。
22 行目：**while** 文を用いて繰り返し処理を行います。
23 行目：**alter** コマンドにより電圧源 **V1** の電圧を上書きします。
24 行目：**ac lin 1 1Meg 1Meg** は AC 解析コマンドで、1 MHz の 1 ポイントのみ解析します。
27〜28 行目：アドミタンスから静電容量を計算し、ベクトル変数 **capacitance** に格納しています。
29 行目：**loop_index** をインクリメントします。
30 行目：**while** 文による繰り返し処理の終了を示します。
32 行目：ベクトル変数 **capacitance** と **reverse_voltage** を関連付けます。
33 行目：結果を **cr-vr.txt** に出力します。
16 行目：**.endc** は制御セクションの終了を示します。
17 行目：**.end** はスクリプト全体の終了を示します。

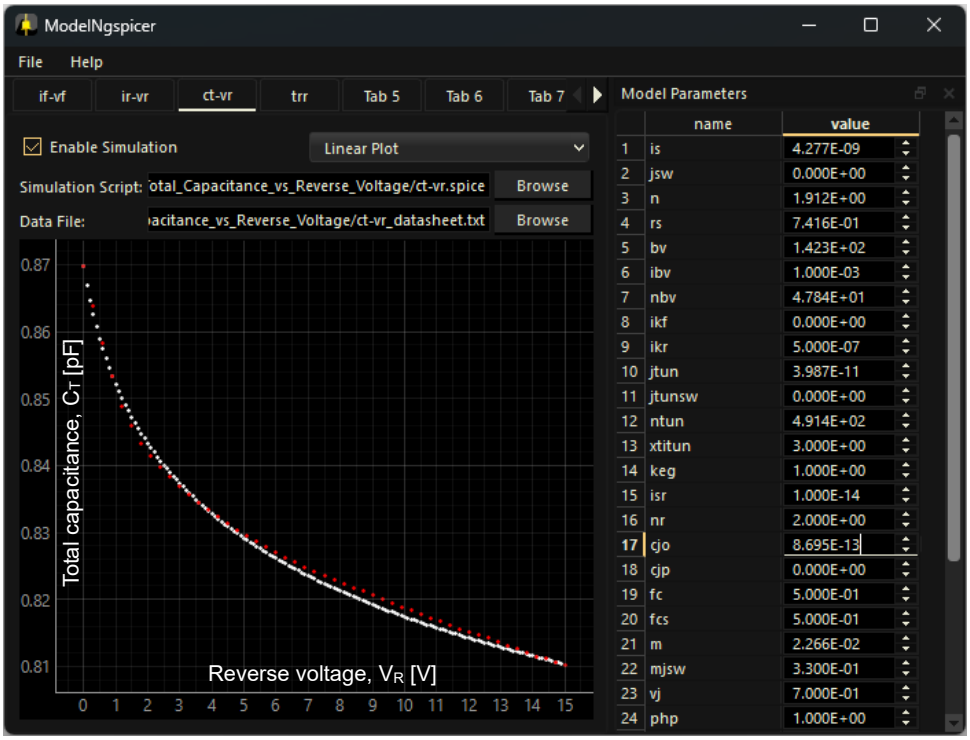

Fig. 5-9　接合容量–逆電圧特性のパラメータ調整
（赤点：データシートの読み値、白点：シミュレーション結果）

モデルパラメータの調整：

ここでは、モデルパラメータ **CJO**（zero-bias junction bottom-wall capacitance）、**M**（area junction grading coefficient）、**VJ**（junction potential）を調整します。

調整後の接合容量–逆電圧特性を Fig. 5-9 に示します。グラフの赤点はデータシートの読み値、白点はシミュレーション結果を示しており、両者はよく一致しています。

5.2.5 逆回復時間

最後に、ダイオードの逆回復時間（t_{rr}）を基にモデルパラメータの調整を行います。逆回復時間とは、ダイオードが順方向から逆方向に切り替わる際、内部に蓄積された過剰キャリア（電子とホール）が抜けきって遮断状態になるまでの遅れ時間のことを指します。

一般的に Fig. 5-10 のような測定回路が用いられます。電圧源 **V2** より順方向電流 I_F（～10 mA）を流した状態から、パルスジェネレータ **V1** より負のパルスを印加し、逆方向電流が収束するまでの時間応答を観測します。

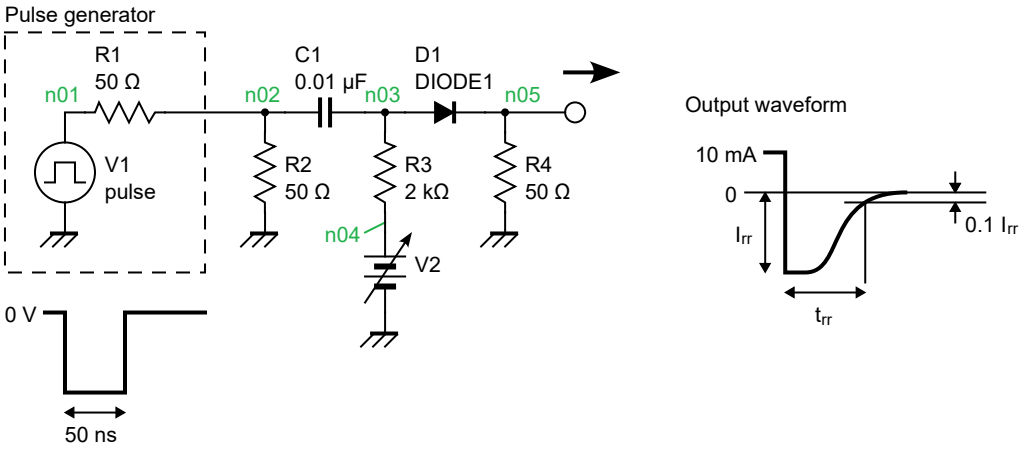

Fig. 5-10　逆回復時間（t_{rr}）の測定回路

ファイル名：**trr.spice**

```
trr.spice

.model DIODE1 D (
.include model.txt
+ )

V1  n01 0   dc 0 pulse(0 -3 10n 0.1n 0.1n 50n 100n 1)
V2  n04 0   dc 21.2
R1  n01 n02 50
R2  n02 0   50
R3  n03 n04 2k
R4  n05 0   50
C1  n02 n03 0.01u
D1  n03 n05 DIODE1

.control
```

```
17 tran 10p 20n
18 wrdata trr.txt v(n05)
19 .endc
20 .end
```

スクリプトの解説：
1 行目：ngspice では 1 行目はコメントとして扱われます（今回はファイル名を記載）。
3〜5 行目：**.model** 文を使用してダイオードモデル **DIODE1** 文を定義しています。**.include** 文によりモデルパラメータを引数として展開します。
7〜14 行目：測定回路をネットリストで記述しています。
16 行目：**.control** により制御セクション（バッチ処理）の開始を示します。
17 行目：**tran 10p 20n** は過渡解析コマンドで、10 psec 刻みで 20 nsec までの時間応答を解析します。
18 行目：結果を **trr.txt** に出力します。
16 行目：**.endc** は制御セクションの終了を示します。
17 行目：**.end** はスクリプト全体の終了を示します。

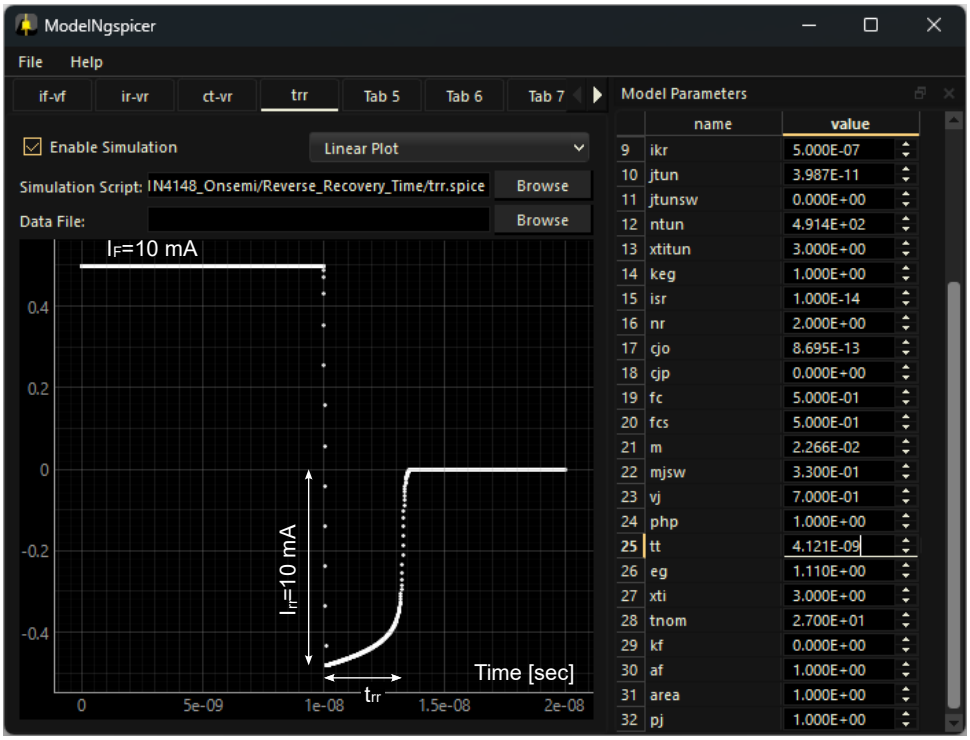

Fig. 5-11　逆回復応答のパラメータ調整

モデルパラメータの調整：
Onsemi 社の 1N4148 のデータシートによると、I_F=10 mA、I_{rr}=10 mA の条件下で、逆回復時間は t_{rr}=3.2 ns とされています。モデルパラメータ **TT**（transit time）を調整することで逆回復特性を合わせていきます。

Fig. 5-11 に調整後の逆回復応答の時間波形を示します。t_{rr}≃3.2 ns となっているのが確認できます。