Hausarbeit

Internationale Hochschule Fernstudium
Studiengang: M. Sc. Informatik

Programmieren mit Python
Visualisierungsmöglichkeiten

B. Eng. Marcin Maślanka
Matrikelnummer:

Betreuungsperson:
Abgabedatum: 01.11.2025

I. Inhaltsverzeichnis

II. Abbildungsverzeichnis

III. Tabellenverzeichnis

1 Einleitung

Menschen haben Zahlen entwickelt, um die Welt auf abstrakte Weise beschreiben zu können. Solange sich diese Zahlen in einem für uns vorstellbaren Bereich befinden, können wir leicht erkennen, was groß, was klein ist usw. Wenn es jedoch um sehr große oder sehr kleine Zahlen geht, wird es schwieriger. In solchen Fällen benötigen wir Hilfsmittel und müssen Vergleiche ziehen.

Ein klassisches Beispiel ist die Vorstellung des Wasserstoffatoms – genauer gesagt des Atomkerns und der Elektronenbahn um den Kern herum. Mit diesen Zahlen sind wir in mathematischer Schreibweise vertraut, aber es fällt uns schwer, sie zu begreifen, wenn wir beispielsweise den Durchmesser des Atomkerns, mit dem der Elektronenbahn vergleichen sollen. Mathematisch betrachtet ist das ein Zahlenwert – aber mehr auch nicht. Um ein besseres Verständnis zu erlangen, benötigen wir eine bildliche Darstellung oder eine Visualisierungshilfe.

Dies lässt sich gut mit einem Gegenstand aus unserer realen Welt veranschaulichen. Ein anschauliches Beispiel ist der Vergleich eines Sandkorns mit einem Fußballstadion. Jeder hat schon einmal ein Sandkorn gesehen und weiß, dass es sehr klein ist. Ebenso kann sich jeder ein Fußballstadion vorstellen. Wenn man sich vorstellt, ein Sandkorn läge in der Mitte des Spielfelds, dann entspräche die Tribüne in diesem Beispiel der Elektronenbahn. Dieses kleine Beispiel öffnet die Augen! Plötzlich erhalten Zahlen im Nanometerbereich eine konkrete Bedeutung und einen Sinn – obwohl wir sie mit bloßem Auge nicht wahrnehmen können. Das ist die Stärke der Visualisierung (*https://www.studysmarter.de/*, 2025, Kapitel Wie groß ist der Atomkern: Ein Überblick).

Für die Visualisierung in Python wurden spezielle Bibliotheken entwickelt, mit denen man große Datenmengen einfach und effizient verarbeiten kann. Zu den bekanntesten zählt Matplotlib, eine weit verbreitete Bibliothek in Python, deren Syntax stark an MATLAB erinnert. Interaktive Visualisierungen lassen sich mit Bokeh und Plotly erstellen. Diese ermöglichen die Integration in HTML-Code und bieten eine kompakte Darstellung mit Zoom-Funktion und automatischer Skalierung. Zudem unterstützen sie die Speicherung in gängigen Bildformaten (McKinney, 2022, S. 244).

Um eine Visualisierung erstellen zu können, benötigen wir Daten. Diese wurden aus dem Programm Ngspice gewonnen, in Excel vorbereiten und anschließend in einer CSV-Datei gespeichert. Damit die Arbeit einen Sinn und eine praktische Bedeutung erhält, wurde eine Designmethodologie vorgestellt, die sich gm/ID oder Transistoreffizienz nennt. Sie ist eine in

der Praxis häufig verwendete Methode zur Konzeption und Analyse elektronischer Schaltungen.

2 Theoretischer Hintergrund

2.1 Die gm/Id-Methode

Die gm/Id-Methode (Transistoreffizienz: TE) findet seit den 1980er-Jahren Anwendung in der analogen Entwurfsmethodik, etwa seit der Ära der Quarz-Armbanduhren. Sie basiert auf dem Verhältnis der Transkonduktanz gm eines MOSFETs zu seinem Drainstrom Id. Diese Methode dient der technologieunabhängigen Charakterisierung und Dimensionierung von MOS-Transistoren in allen Inversionsbereichen (schwache, moderate und starke Inversion) (Oehme et al., 2025, S. 228).

Im Gegensatz zur klassischen Square-Law-Approximation (die nur in starker Inversion gilt), erlaubt die gm/Id-Methode mithilfe von Simulationsdaten eine präzise Erfassung von Bauelementeigenschaften, wie beispielsweise der Dimensionierung von W (Breite) und L (Länge) des Transistors.

Die Transkonduktanz ist mathematisch definiert als die Ableitung des Drainstroms nach der Gate-Source-Spannung:

$$gm = \frac{d\,Id}{d\,Vgs} \qquad \text{Gl. 1}$$

Das zentrale Verhältnis der Methode lautet:

$$TE = \frac{gm}{Id} \qquad \text{Gl. 2}$$

Dieses Verhältnis beschreibt die Effizienz eines Transistors, d. h. wie viel Strom Id muss eingesetzt werden, um auf eine Transkonduktanz gm zu gelangen. Die Einheit ist 1/V.

Typische Werte:

- Schwache Inversion (Subthreshold): 25–30 1/V
- Starke Inversion: 5–10 1/V

Somit kann der Designer anhand des gewünschten gm/Id-Werts den optimalen Betriebsbereich des Transistors auswählen – je nach Priorität zwischen Verstärkung oder Geschwindigkeit.

2.2 Datenauswertung

Ngspice ist ein Open-Source-Schaltungssimulationsprogramm, es ermöglicht den Export von Simulationsergebnissen in verschiedenen Formaten wie .raw oder .txt. Für die Weiterverarbeitung mit Python – insbesondere mit Pandas – ist das CSV-Format die bevorzugte Wahl, da es einfach und transparent ist. Das CSV-Format (Comma Separated Values) ist eine textbasierte Tabellenstruktur, bei der jede Zeile einen Datensatz repräsentiert und die einzelnen Werte durch ein Trennzeichen (Komma) voneinander getrennt sind. Eine Kopfzeile definiert die Spaltennamen (Matthes, 2023, S. 392,393).

Die Bibliothek Pandas ist ein Standardwerkzeug für die Datenanalyse in Python. Sie erlaubt das komfortable Einlesen und Verarbeiten von CSV-Dateien mit wenigen Codezeilen:

Tabelle 1 Import Pandas

```
1  import pandas as pd
2  file_path =
3  "/home/marcin/Dokumente/PY/Projekt2/PY/test_gmid_130_500_1000.csv"
   data = pd.read_csv(file_path)
```

2.3 Relevante Python-Bibliotheken

Für die Analyse und Visualisierung kommen insbesondere drei Bibliotheken zum Einsatz: NumPy, Pandas und Matplotlib.

2.3.1 NumPy

NumPy (Numerical Python) ist eine Bibliothek für numerische Berechnungen. Sie führt das Array-Konzept in Python ein – eine Datenstruktur, die Matrizen in MATLAB ähnelt.

Beispiel für die numerische Ableitung:

Tabelle 2 Import NumPy

```
1 import numpy as np
2 vgs = np.array([0.1, 0.2, 0.3])
3 id = np.array([1.2e-8, 3.5e-8, 7.9e-8])
4 gm= np.gradient(id, vgs)
```

2.3.2 Pandas

Pandas erweitert NumPy um eine tabellenartige Datenstruktur, den sogenannten Data Frame. Dieser ist vergleichbar mit einer Excel-Tabelle und erlaubt eine intuitive Handhabung von Mess- und Simulationsdaten.

Tabelle 3 Excel Tabelle

Vgs	Id	gm	gds	cgg	vth	Length
0	2.212262E-11	5.981323E-10	1.8092E-11	9.881095E-17	0.4551296	1.30E-07
0.1	3.389178E-10	0.000000009298621	2.567196E-10	9.268518E-17	0.4551296	1.30E-07
0.2	0.000000005335003	0.000000147597 7	0.000000004148737	8.78162E-17	0.4551296	1.30E-07
…	…	…	…	…	…	…
0	2.194364E-10	0.000000006791806	1.178187E-10	7.141944E-16	0.2895441	5.00E-07
0.1	0.000000004907297	0.000000152837 6	0.000000002 69864	6.85826E-16	0.2895441	5.00E-07
0.2	0.000000106799	0.000003180401	0.000000059 81948	7.741835E-16	0.2895441	5.00E-07
…	…	…	…	…	…	…

2.3.3 Matplotlib

Matplotlib ist die Standardbibliothek für wissenschaftliche Visualisierung in Python. Sie ermöglicht die Erstellung von 2D-Diagrammen mit detaillierter Kontrolle über Achsen, Farben und Beschriftungen.

Die Darstellung mehrere Kurven in einem Diagramm erfordert eine klare Unterscheidung durch Farben. Dafür stehen verschiedene Colormaps zur Verfügung. Standardmäßig ist in Matplotlib die Colormap tab10 voreingestellt, die sich gut für die Darstellung von bis zu 10 unterschiedlichen Kurven eignet.

Diese Farben sind so gewählt, dass sie einerseits leicht unterscheidbar sind und anderseits keinen falschen Eindruck vermitteln -etwa, dass eine Kurve dominierter oder wichtiger als die anderen erscheint. Dadurch wird eine neutrale und ausgewogene Darstellung der Daten gewährleistet (Wilke, 2025, Kapitel 4.1 Color as a tool to distinguish).

Tabelle 4 Matplotlib

```
1 plt.plot(vgs, id_, label=f'L = {L*1e9:.0f} nm', linewidth=1.8)
2 plt.xlabel("Vgs (V)")
3 plt.ylabel("Id (A)")
4 plt.title("Input Charakteristics")
5 plt.grid(True)
6 plt.legend()
7 plt.tight_layout()
8 plt.show()
```

3 Methodik und Implementierung

Zur Untersuchung wurde ein NMOS-Transistor aus dem Process-Design-Kit IHP-SG13G2 mit einer Strukturgröße von 130 nm ausgewählt. Der zugehörige Schaltplan ist unten dargestellt. Neben dem Transistor sind zwei Spannungsquellen eingebunden. Eine davon ist zwischen Gate und Source angeschlossen und wird gesweept, das heißt, die Spannung wird schrittweise um 0,1 V erhöht, bis eine bestimmte Maximalspannung erreicht ist. Die zweite Spannungsquelle ist am Ausgang angeschlossen und beträgt etwa die Hälfte der gesamten Versorgungsspannung.

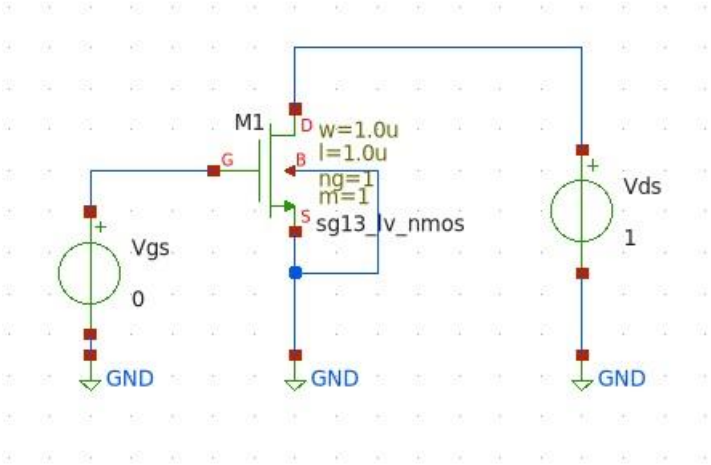


Abbildung 1 Schaltplan

Diese Konfiguration erlaubt die Untersuchung der Transistorparameter in Abhängigkeit von der Eingangsspannung U_{GS} und der Kanallänge, die als zweiter Parameter in drei Stufen variiert wird: beginnend mit 130 nm, über 500 nm bis hin zu 1000 nm.

Mithilfe der Ngspice-Simulationssoftware können die gewünschten Transistorparameter in einer Textdatei gespeichert werden (im Anhang 7.3). Diese Datei ist jedoch nicht direkt kompatibel mit Python, weshalb eine weitere Vorbereitung notwendig ist. Ziel ist es, eine CSV-Datei zu erstellen, mit der sich die einzelnen Transistorparameter einfach auslesen und weiterverarbeiten lassen.

Die Vorbereitung der Textdatei erfolgt mit Microsoft Excel, wobei die einzelnen Spalten hintereinander angeordnet werden. Am Ende jeder Messreihe wird zusätzlich die Kanallänge des Transistors ergänzt. Auf diese Weise entsteht eine CSV-Datei (im Anhang 7.2) mit sieben Spalten und mehreren Zeilen, wobei drei Gruppen von Zeilen erkennbar sind – jeweils für eine der drei Kanallängen. Eine CSV-Datei beginnt mit einem Header, der die Bedeutung der einzelnen Spalten beschreibt.

Zur Visualisierung wurde die Jupyter-Notebook-Umgebung verwendet, die speziell für Datenanalyse konzipiert ist und eine komfortable Ausführung einzelner Codeabschnitte ermöglicht. Im ersten Schritt werden die benötigten Bibliotheken importiert:

- Pandas zum Einlesen der CSV-Datei
- NumPy für die Datenanalyse
- Matplotlib für die Visualisierung

Die CSV-Datei wird mit Pandas eingelesen und in einer Variable namens data gespeichert. Als Probe wurde mit dem Befehl print() ein Ausdruck erzeugt, der die Daten zeilenweise korrekt darstellt.

Im nächsten Schritt soll ein Plot der Eingangscharakteristik des Transistors für die drei Kanallängen erstellt werden. Dazu wurden aus der Spalte Length mithilfe der Methode unique() die vorhandenen Kanallängen extrahiert und in einer Variable lengths gespeichert. Diese Variable wird in einer for-Schleife verwendet, in der die jeweiligen Parameter ausgelesen, temporär gespeichert, mathematisch verarbeitet und schließlich mit dem Befehl plot() dargestellt werden. Der Graph entsteht aus einzelnen Punkten, die nach Abschluss der Schleife ein vollständiges Diagramm bilden.

Außerhalb der Schleife werden Plot-Parameter hinzugefügt, um Achsen, Titel und Legende darzustellen. Am Ende entsteht ein ansprechender Graph mit den drei Eingangscharakteristiken für die verschiedenen Kanallängen.

Die Eingangscharakteristik dient als erste Probe. Die eigentliche Kurve ist jedoch der Graph von gm/Id in Abhängigkeit von der Overdrive-Spannung V_{ov}. Dies lässt sich leicht umsetzen, indem man die Argumente des plot()-Befehls entsprechend anpasst und die gewünschten Parameter als x- und y-Koordinaten verwendet.

Um die gm/Id-Methode vollständig nutzen zu können, sind weitere Graphen erforderlich. Dazu gehören:

- Transitfrequenz
- Intrinsische Verstärkung
- Combined Performance Metric

Diese sollen mithilfe von Subplots dargestellt werden. Insgesamt werden vier Subplots benötigt, um die vollständige gm/Id-Methodologie zu visualisieren. Dabei müssen die Achsenbeschriftungen und Titel für jeden einzelnen Subplot innerhalb der Schleife definiert werden.

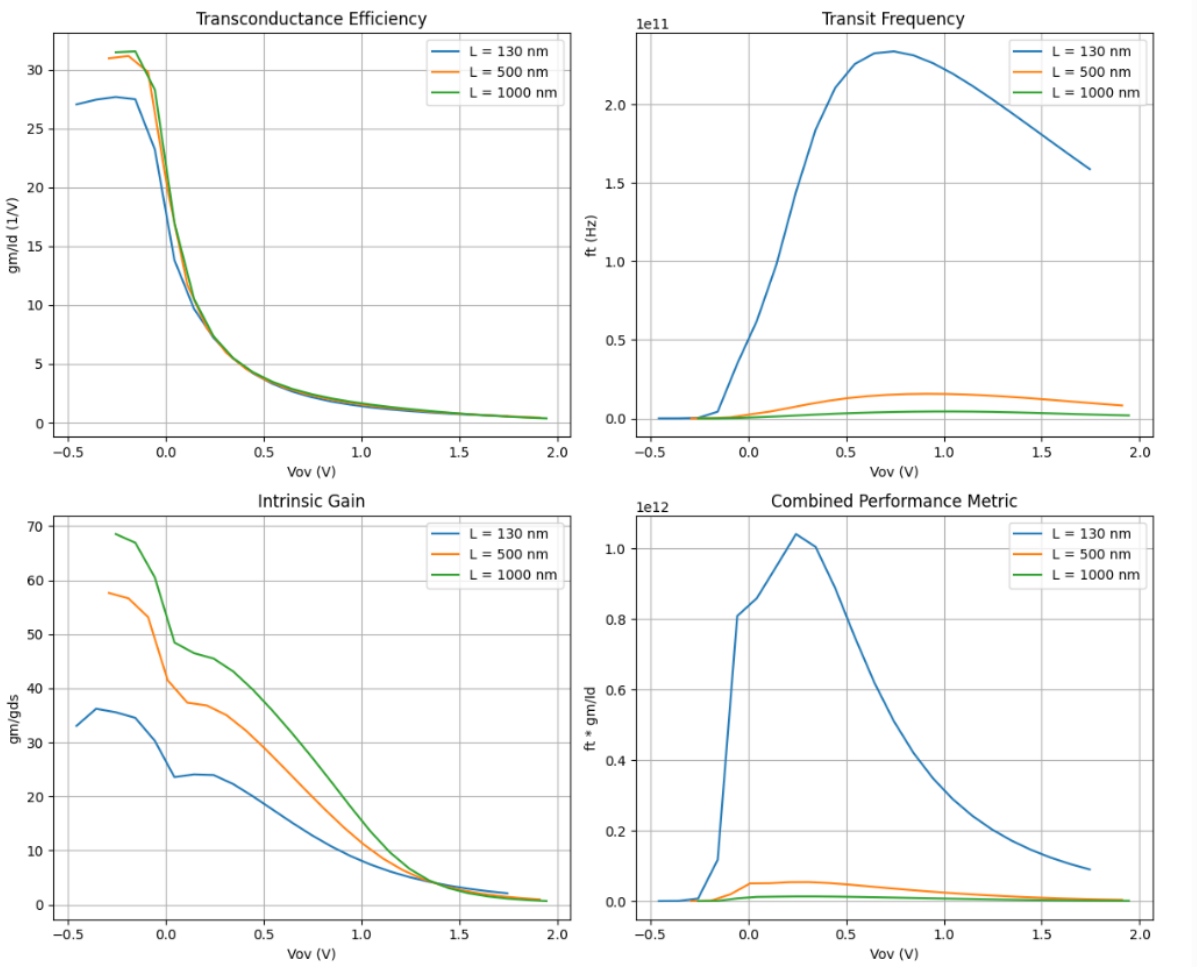


Abbildung 2 4-Graphen Chart im Matplotlib

Eine interessante Möglichkeit zur Visualisierung bietet das Hinzufügen eines weiteren Parameters. Ein Beispiel ist der Plot von gm/Id in Abhängigkeit von V_{ov}, wobei die Frequenz als zusätzlicher Parameter dargestellt wird. Dieser Plot ergibt physikalisch Sinn und entspricht der Theorie: Ein Transistor in starker Inversion wird mit hohen Frequenzen betrieben, während in schwacher Inversion nur niedrige Betriebsfrequenzen möglich sind. Ein Scatter-Plot mit farblicher Darstellung eignet sich hervorragend für diese Visualisierung (Wilke, 2025, Kapitel 2.2 Scales map data values onto aesthetics).

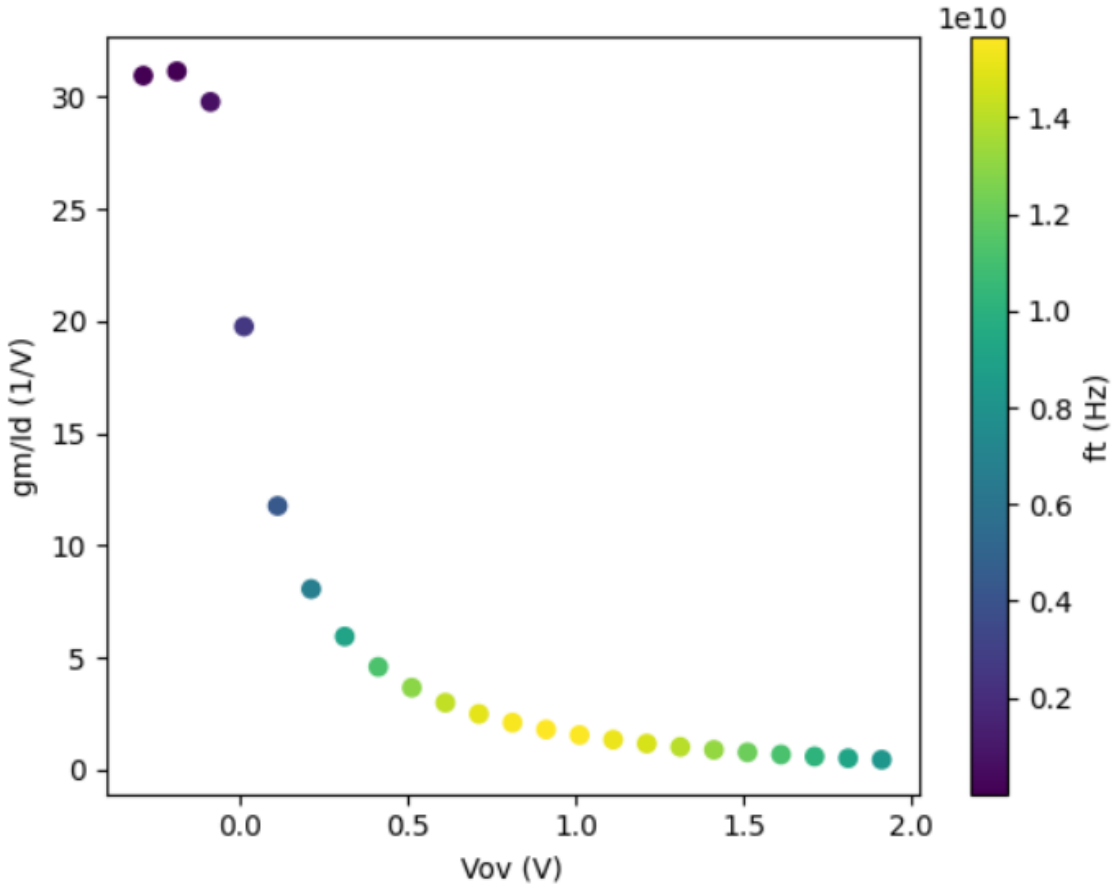


Abbildung 3 zweites Argument in der Funktion gm/id(vov, ft)

In der gm/Id-Methode ist es notwendig, Werte direkt aus dem Graphen auszulesen. Um dies zu ermöglichen, wurden die externen Bibliotheken untersucht: Bokeh und Plotly.

Wie bereits bei der Visualisierung mit Matplotlib erwähnt, bestehen die Plots aus einzelnen Punkten. Die untersuchten Bibliotheken erzeugen eine Approximation zwischen den Datenpunkten und zeichnen eine durchgehende Verbindungslinie (Wilke, 2025, Kapitel 13.1 Individual time series). Als Anwender sind wir jedoch oft an beliebigen Punkten zwischen den Messwerten interessiert. Dies war meine Motivation, herauszufinden, welche Bibliothek dies am einfachsten ermöglicht.

Mit der Bokeh-Bibliothek lassen sich interaktive Graphen erstellen. Besonders hervorzuheben ist die Möglichkeit, den Graphen mit dem Mausrad zu zoomen und zu verschieben. Zudem bietet die Bibliothek eine Werkzeugleiste mit häufig genutzten Funktionen. Falls man die Orientierung im Diagramm verliert, kann man mit einem Reset-Button zur Ausgangsposition zurückkehren. Die Skalierung passt sich automatisch an, sodass man stets die aktuellen Werte ablesen kann. Allerdings gibt es keine Live-Anzeige der x- und y-Koordinaten. Diese Funktion

bietet Plotly. Die Bibliothek ist Bokeh sehr ähnlich, bietet jedoch zusätzlich die Möglichkeit, aktuelle Werte in Form einer Legende anzuzeigen. Dies ist jedoch nicht exakt das, was angestrebt wurde – keine der Bibliotheken stellt approximierte Werte direkt zur Verfügung. Man kann zwar die Dichte der Datenpunkte erhöhen, um genauere Werte zu erhalten, oder ein kurzes Python-Skript schreiben, um gezielt auf bestimmte Punkte zuzugreifen. Dies wäre jedoch aufwendig, da jedes Mal ein neues Skript erstellt und angepasst werden müsste.

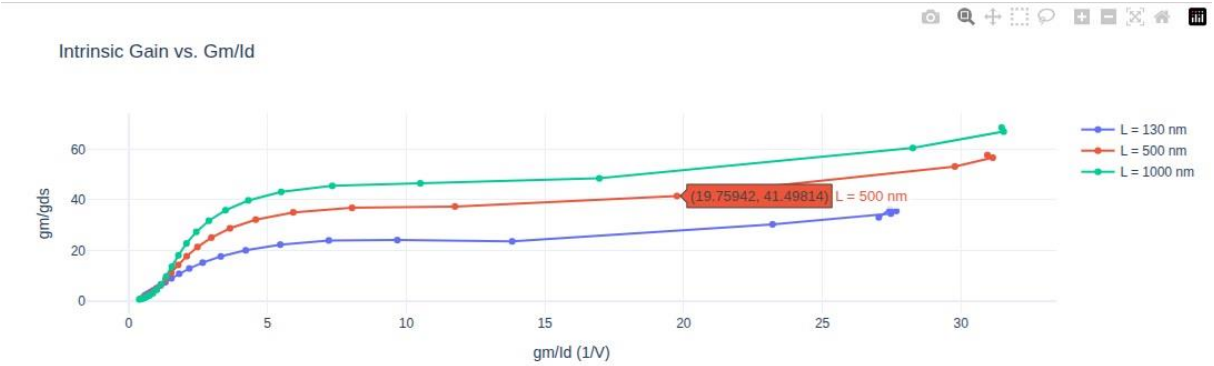


Abbildung 4 Interactive Plot mit Plotly

Es scheint daher am einfachsten zu sein, die Autoskalierung und Zoom-Funktion zu nutzen, um gezielt an bestimmte Werte heranzukommen.

4 Ergebnisse

Zur quantitativen Bewertung der Kurve gm/Id(V_{ov}) werden drei statistische Größen bestimmt: Mittel-, Minimal- und Maximalwert.

Tabelle 5 Berechnung von Mittel-, Minimal- und Maximalwerten:

```
1 mean_gm_id = data[„gm_id“].mean()
2 min_gm_id = data[„gm_id“].min()
3 max_gm_id = data[“gm_id”].max()
```

Diese Werte werden typischerweise mit dem Befehl print() ausgegeben oder direkt im Plot dargestellt. Sie besitzen eine physikalische Bedeutung, die mit den oben erwähnten Inversionsstufen des Transistors zusammenhängen:

- Der Maximalwert entspricht dem Bereich der schwachen Inversion, in dem eine hohe Verstärkung bei niedriger Geschwindigkeit erzielt wird.
- Der Minimalwert steht für den Bereich der starken Inversion, der durch geringe Verstärkung und hohe Geschwindigkeit gekennzeichnet ist.
- Der Mittelwert wird in der Praxis am häufigsten als Betriebsbereich verwendet, da er einen Kompromiss zwischen hoher Verstärkung und hoher Geschwindigkeit bietet.

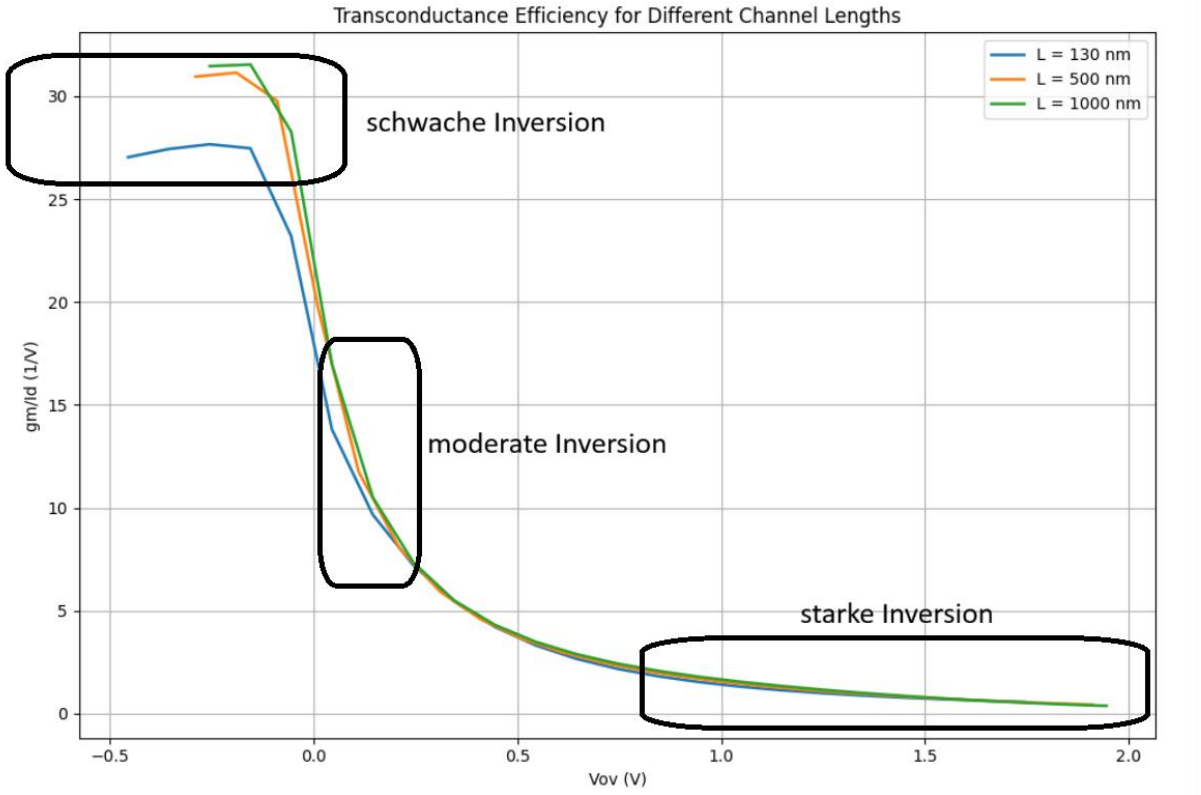


Abbildung 5 Inversionsbereiche

Die durchgeführte Auswertung zeigt, dass die gm/Id-Methode nicht nur qualitative, sondern auch quantitative Einblicke in das Verhalten von MOS-Transistoren ermöglicht. Durch die Berechnung von Durchschnitts-, Minimal- und Maximalwerten des Verhältnisses gm/Id konnten charakteristische Effizienzbereiche identifiziert werden.

Mit der Methode errorbar() wurde die Abweichung von der gm/Id Kurve von der Quadratischen Gleichung dargestellt (Anhang 7.4). Im zugehörigen Diagramm ist erkennbar, dass der Fehlerbalken ab einem bestimmten Punkt auf der x-Achse -etwa 0,2V deutlich zunimmt. Dieses Verhalten bestätigt, dass die quadratische Gleichung nur im Bereich der starken Inversion gültig ist und nicht auf die moderate oder schwache Inversion anwendbar ist.

Zusätzlich wurde eine Heatmap erstellt, um den Einfluss der geometrischen Transistorparameter auf die Frequenz zu visualisieren. Die Länge zwischen Source und Drain

eines MOSFETs hat einen starken Einfluss auf die Geschwindigkeit, während die Breite nur einen linearen Zusammenhang zeigt. Dies ergibt sich aus der vereinfachten quadratischen Gleichung:

$$Id \infty \frac{W}{L} \qquad \text{Gl. 3}$$

Dieser Zusammenhang lässt sich grafisch in einer Heatmap darstellen, bei der die x-Achse die Länge, die y-Achse die Breite des Transistors zeigt. Die Farbskala repräsentiert die Frequenz in GHz (Anhang 7.5).

5 Kritische Betrachtung

Das automatische Einlesen von Daten kann potenziell auch Probleme mit sich bringen. Diese entstehen häufig durch Fehler in der CSV-Datei, wie zum Beispiel falsche Dezimaltrennzeichen – etwa Komma statt Punkt. Auch fehlende numerische Werte stellen ein typisches Problem dar, das in der Praxis auftreten kann. Solche Fehler entstehen beispielsweise durch Unterbrechungen im Messvorgang oder Inkonsistenzen zwischen verschiedenen Simulationen, bei denen eine Spalte zu viel oder zu wenig Daten enthält als eine andere (Matthes, 2023, S. 402).

Ein weiteres häufiges Problem in der Praxis sind unterschiedliche Einheiten. Diese können zu fehlerhaften Interpretationen führen und letztlich zu falschen Ergebnissen. Grundsätzlich empfiehlt es sich, mit den sieben Basiseinheiten zu arbeiten und die wissenschaftliche Schreibweise zu verwenden, um Einheitlichkeit zu gewährleisten (Wilke, 2025, Kapitel 3.1 Cartesian coordinates).

Vor Beginn einer Datenanalyse lohnt es sich stets zu überprüfen, ob die extrahierten Daten sinnvoll und konsistent sind. Aus diesem Grund habe ich zu Beginn ein einfaches Plot erstellt – die Eingangscharakteristik des NMOS-Transistors. Erwartungsgemäß wurde eine quadratische Kurve dargestellt, was dem theoretischen Verhalten entspricht. Die drei unterschiedlichen Kurven lassen sich ebenfalls nachvollziehen: Je länger der Kanal des Transistors, desto kleiner der Strom. Damit wurde ein Plausibilitätsprüfung durchgeführt, und die eigentliche Analyse kann beginnen.

In diesem konkreten Beispiel ist die Validierung besonders einfach, da die Daten aus dem Simulationsprogramm Ngspice stammen. Man kann den erzeugten Plot direkt mit dem

Diagramm aus Ngspice vergleichen, um sicherzustellen, dass die Daten korrekt erfasst wurden.

Hier stellt sich die berechtigte Frage: Warum der Umweg über die Visualisierung mit Python? Könnte man nicht direkt in Ngspice die gm/Id-Kurven erzeugen?

Die Antwort lautet: Ja und nein.

Ja, weil es theoretisch möglich ist, die gewünschten Kurven direkt in Ngspice zu generieren. Nein, weil die Simulation in Ngspice sehr zeitaufwendig ist. Stattdessen wird eine LUT (Lookup-Tabelle) erstellt, um die Daten aus dem Simulationsprogramm zu extrahieren. Mit dieser einmal generierten LUT kann man dann schnell und flexibel weiterarbeiten, ohne Ngspice erneut ausführen zu müssen. Dadurch wird das Simulationsprogramm aus dem Workflow entfernt, was die Analyse deutlich beschleunigt.

6 Zusammenfassung

In dieser Arbeit wurde gezeigt, wie Visualisierungstechniken in Python zur Analyse technischer Messdaten eingesetzt werden können. Am Beispiel der gm/Id-Methode aus der analogen Schaltungstechnik wurden die Transistordaten in CSV-Dateien überführt und mit Bibliotheken wie Pandas, NumPy und Matplotlib analysiert. Die Visualisierung ermöglichte nicht nur die Darstellung klassischer Eingangscharakteristiken, sondern auch komplexer Kennlinien wie Transitfrequenz und intrinsische Verstärkung, die eine praktische Bedeutung für analog Designer haben.

Besonderes Augenmerk wurde auf die Fehlerquellen beim automatischen Einlesen gelegt. Vor dem Visualisierungsverfahren wurde einen Plausibilitätsprüfung der Daten durchgeführt, dies könnte beispielsweise der Eingangskennlinie der Transistoren sein. Damit ist die Zuverlässigkeit der Daten sichergestellt.

Die Arbeit zeigte außerdem die Vorteile interaktiver Visualisierung mit Bokeh und Plotly, insbesondere bei der herausfinden der Werten bei der Approximation. Trotz der Einschränkungen bei der direkten Auslesung von Zwischenwerten bieten diese Tools Möglichkeiten zu Zoomen und die Auto-Skalierung.

Zur quantitativen Bewertung wurden Durchschnitts-, Minimal- und Maximalwerte berechnet, die eine eindeutige Zuordnung zu verschiedenen Inversionsbereichen des Transistors

erlauben. Diese Kennwerte konnten mithilfe von Diagrammen intuitiv erfasst werden, was den Vorteil der Visualisierung gegenüber reinen Zahlenwerten unterstreicht.

7 Anhänge

7.1 Github Repository

Der vollständige Python-Quellcode, die verwendeten CSV-Dateien sowie die Jupyter-Notebook Implementierung sind unter folgendem Repository abrufbar:

https://github.com/marcinmaslanka/PY/tree/master

7.2 CSV-Datei

```
Vgs,Id,gm,gds,cgg,vth,Length
0,2.212262E-11,5.981323E-10,1.8092E-11,9.881095E-17,0.4551296,1.30E-07
0.1,3.389178E-10,0.000000009298621,2.567196E-10,9.268518E-17,0.4551296,1.30E-07
0.2,0.000000005335003,0.0000001475977,0.000000004148737,8.78162E-
17,0.4551296,1.30E-07
0.3,0.00000008474021,0.000002327827,0.00000006738317,8.672333E-
17,0.4551296,1.30E-07
0.4,0.000001145621,0.00002658691,0.000000877718,1.214487E-16,0.4551296,1.30E-07
0.5,0.000007070476,0.00009771385,0.000004144727,2.501473E-16,0.4551296,1.30E-07
0.6,0.00002229824,0.0002158461,0.000008963905,3.501775E-16,0.4551296,1.30E-07
0.7,0.00005150018,0.000371223,0.00001549219,4.091196E-16,0.4551296,1.30E-07
0.8,0.0000965445,0.0005270959,0.00002363594,4.558733E-16,0.4551296,1.30E-07
0.9,0.0001560533,0.0006575442,0.0000328028,4.96676E-16,0.4551296,1.30E-07
1,0.0002268549,0.0007524409,0.00004272472,5.301558E-16,0.4551296,1.30E-07
1.1,0.0003054057,0.0008134274,0.00005359855,5.562177E-16,0.4551296,1.30E-07
1.2,0.0003886366,0.000847303,0.00006582324,5.763766E-16,0.4551296,1.30E-07
1.3,0.0004742047,0.0008612753,0.00007982054,5.92322E-16,0.4551296,1.30E-07
1.4,0.0005604234,0.0008611454,0.00009596196,6.053742E-16,0.4551296,1.30E-07
1.5,0.0006461054,0.0008511235,0.0001145392,6.164539E-16,0.4551296,1.30E-07
1.6,0.0007304176,0.0008341572,0.0001357528,6.261729E-16,0.4551296,1.30E-07
1.7,0.0008127746,0.0008123031,0.0001597109,6.349258E-16,0.4551296,1.30E-07
1.8,0.0008927643,0.0007870166,0.0001864345,6.429605E-16,0.4551296,1.30E-07
1.9,0.000970099,0.0007593517,0.0002158661,6.504282E-16,0.4551296,1.30E-07
2,0.001044582,0.0007300915,0.0002478816,6.574183E-16,0.4551296,1.30E-07
2.1,0.001116084,0.0006998326,0.0002823029,6.639815E-16,0.4551296,1.30E-07
2.2,0.001184531,0.0006690387,0.0003189106,6.701459E-16,0.4551296,1.30E-07
0,2.194364E-10,0.000000006791806,1.178187E-10,7.141944E-16,0.2895441,5.00E-07
0.1,0.000000004907297,0.0000001528376,0.00000000269864,6.85826E-
16,0.2895441,5.00E-07
0.2,0.000000106799,0.000003180401,0.00000005981948,7.741835E-
16,0.2895441,5.00E-07
0.3,0.000001365118,0.00002697394,0.0000006500036,1.688366E-15,0.2895441,5.00E-
07
0.4,0.000006233708,0.00007328544,0.000001961871,2.705197E-15,0.2895441,5.00E-07
0.5,0.00001645579,0.0001324355,0.000003596425,3.163214E-15,0.2895441,5.00E-07
0.6,0.00003281686,0.0001945926,0.00000555276,3.411517E-15,0.2895441,5.00E-07
0.7,0.00005523681,0.0002527308,0.00000785538,3.587608E-15,0.2895441,5.00E-07
0.8,0.00008309763,0.0003029837,0.00001054282,3.732996E-15,0.2895441,5.00E-07
0.9,0.0001155165,0.0003437722,0.00001371123,3.860772E-15,0.2895441,5.00E-07
1,0.0001515306,0.0003749376,0.00001755678,3.975368E-15,0.2895441,5.00E-07
1.1,0.0001902011,0.0003970209,0.00002241676,4.079008E-15,0.2895441,5.00E-07
1.2,0.0002306574,0.0004107823,0.0000287982,4.173912E-15,0.2895441,5.00E-07
1.3,0.000272105,0.0004169635,0.00003736791,4.262852E-15,0.2895441,5.00E-07
1.4,0.00031382,0.0004162397,0.00004887768,4.348864E-15,0.2895441,5.00E-07
1.5,0.000355145,0.0004092787,0.00006402461,4.434566E-15,0.2895441,5.00E-07
1.6,0.0003954926,0.0003968256,0.00008328628,4.521477E-15,0.2895441,5.00E-07
1.7,0.000434356,0.0003797492,0.0001067936,4.609722E-15,0.2895441,5.00E-07
1.8,0.0004713211,0.0003590269,0.0001342895,4.698209E-15,0.2895441,5.00E-07
1.9,0.0005060742,0.0003356812,0.0001651797,4.785139E-15,0.2895441,5.00E-07
2,0.0005384032,0.0003106992,0.0001986454,4.868573E-15,0.2895441,5.00E-07
2.1,0.0005681896,0.0002849635,0.0002337732,4.94685E-15,0.2895441,5.00E-07
2.2,0.0005953961,0.0002592098,0.0002696668,5.018807E-15,0.2895441,5.00E-07
0,2.531049E-10,0.00000000796362,1.162077E-10,1.51934E-15,0.2547715,1.00E-06
```

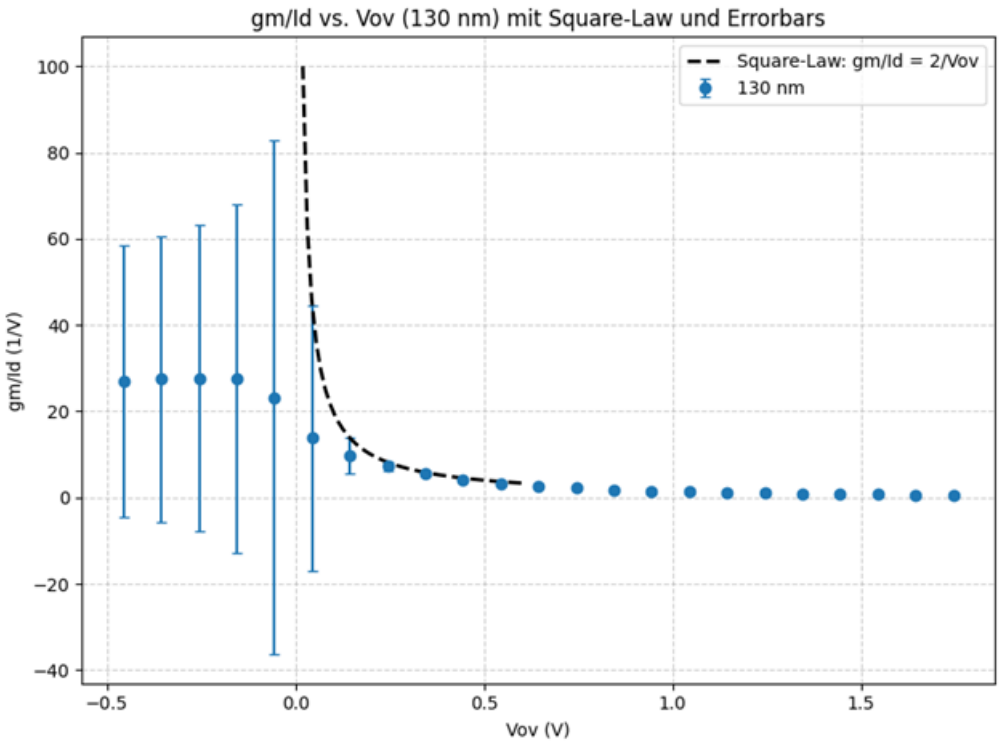
0.1,0.000000005937884,0.0000001872725,0.000000002798291,1.48261E-15,0.2547715,1.00E-06
0.2,0.0000001263815,0.000003572002,0.00000005906333,2.058386E-15,0.2547715,1.00E-06
0.3,0.000001214388,0.0000205984,0.0000004249354,4.660402E-15,0.2547715,1.00E-06
0.4,0.000004595337,0.0000482896,0.000001038434,6.376068E-15,0.2547715,1.00E-06
0.5,0.00001103627,0.00008094489,0.000001778935,7.125109E-15,0.2547715,1.00E-06
0.6,0.00002080549,0.0001142924,0.000002650528,7.543814E-15,0.2547715,1.00E-06
0.7,0.00003382873,0.0001456974,0.000003664135,7.836126E-15,0.2547715,1.00E-06
0.8,0.00004983024,0.0001736854,0.000004840002,8.069989E-15,0.2547715,1.00E-06
0.9,0.0000684274,0.0001975358,0.000006227542,8.272425E-15,0.2547715,1.00E-06
1,0.00008919025,0.0002169781,0.00000793828,8.456854E-15,0.2547715,1.00E-06
1.1,0.000111674,0.0002319493,0.00001019777,8.63208E-15,0.2547715,1.00E-06
1.2,0.0001354297,0.0002424037,0.00001341887,8.805626E-15,0.2547715,1.00E-06
1.3,0.0001599993,0.0002481884,0.00001828104,8.985466E-15,0.2547715,1.00E-06
1.4,0.0001849026,0.0002490283,0.00002575004,9.180934E-15,0.2547715,1.00E-06
1.5,0.0002096319,0.0002446855,0.00003691003,9.401169E-15,0.2547715,1.00E-06
1.6,0.00023367,0.0002352673,0.00005254334,9.649888E-15,0.2547715,1.00E-06
1.7,0.000256539,0.0002214687,0.00007266852,9.920419E-15,0.2547715,1.00E-06
1.8,0.0002778585,0.000204508,0.00009642069,1.019708E-14,0.2547715,1.00E-06
1.9,0.0002973827,0.0001857916,0.0001223804,1.046228E-14,0.2547715,1.00E-06
2,0.0003150009,0.0001665679,0.0001490638,1.070298E-14,0.2547715,1.00E-06
2.1,0.000330711,0.0001477508,0.0001752639,1.091279E-14,0.2547715,1.00E-06
2.2,0.0003445849,0.0001299105,0.0002001619,1.109079E-14,0.2547715,1.00E-06

7.3 TXT-Datei aus dem Ngspice

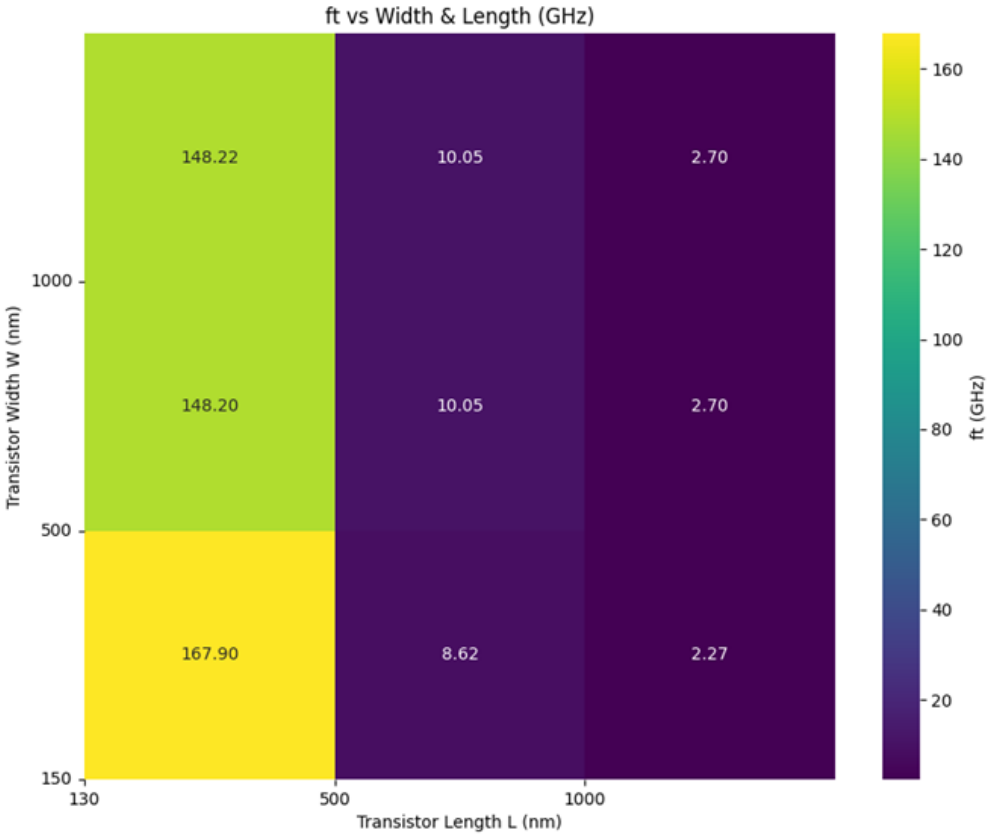
```
            ** sch_path: /foss/designs/ihp/nmos/nmos_tb.sch
            DC transfer characteristic  Sun Oct 19 16:36:33  2025
--------------------------------------------------------------------------------
Index   v-sweep        @n.xm1.nsg13_lv @n.xm1.nsg13_lv @n.xm1.nsg13_lv
--------------------------------------------------------------------------------
0       0.000000e+002.212262e-11 5.981323e-10 1.809200e-11
1       1.000000e-01 3.389178e-10 9.298621e-09 2.567196e-10
2       2.000000e-01 5.335003e-09 1.475977e-07 4.148737e-09
3       3.000000e-01 8.474021e-08 2.327827e-06 6.738317e-08
4       4.000000e-01 1.145621e-06 2.658691e-05 8.777180e-07
5       5.000000e-01 7.070476e-06 9.771385e-05 4.144727e-06
6       6.000000e-01 2.229824e-05 2.158461e-04 8.963905e-06
7       7.000000e-01 5.150018e-05 3.712230e-04 1.549219e-05
8       8.000000e-01 9.654450e-05 5.270959e-04 2.363594e-05
9       9.000000e-01 1.560533e-04 6.575442e-04 3.280280e-05
10      1.000000e+002.268549e-04 7.524409e-04 4.272472e-05
11      1.100000e+003.054057e-04 8.134274e-04 5.359855e-05
12      1.200000e+003.886366e-04 8.473030e-04 6.582324e-05
13      1.300000e+004.742047e-04 8.612753e-04 7.982054e-05
14      1.400000e+005.604234e-04 8.611454e-04 9.596196e-05
15      1.500000e+006.461054e-04 8.511235e-04 1.145392e-04
16      1.600000e+007.304176e-04 8.341572e-04 1.357528e-04
17      1.700000e+008.127746e-04 8.123031e-04 1.597109e-04
18      1.800000e+008.927643e-04 7.870166e-04 1.864345e-04
19      1.900000e+009.700990e-04 7.593517e-04 2.158661e-04
20      2.000000e+001.044582e-03 7.300915e-04 2.478816e-04
21      2.100000e+001.116084e-03 6.998326e-04 2.823029e-04
22      2.200000e+001.184531e-03 6.690387e-04 3.189106e-04

            ** sch_path: /foss/designs/ihp/nmos/nmos_tb.sch
            DC transfer characteristic  Sun Oct 19 16:36:33  2025
--------------------------------------------------------------------------------
Index   v-sweep        @n.xm1.nsg13_lv @n.xm1.nsg13_lv
--------------------------------------------------------------------------------
0       0.000000e+009.881095e-17 4.551296e-01
1       1.000000e-01 9.268518e-17 4.551296e-01
2       2.000000e-01 8.781620e-17 4.551296e-01
3       3.000000e-01 8.672333e-17 4.551296e-01
4       4.000000e-01 1.214487e-16 4.551296e-01
5       5.000000e-01 2.501473e-16 4.551296e-01
6       6.000000e-01 3.501775e-16 4.551296e-01
7       7.000000e-01 4.091196e-16 4.551296e-01
8       8.000000e-01 4.558733e-16 4.551296e-01
9       9.000000e-01 4.966760e-16 4.551296e-01
10      1.000000e+005.301558e-16 4.551296e-01
11      1.100000e+005.562177e-16 4.551296e-01
12      1.200000e+005.763766e-16 4.551296e-01
13      1.300000e+005.923220e-16 4.551296e-01
14      1.400000e+006.053742e-16 4.551296e-01
15      1.500000e+006.164539e-16 4.551296e-01
16      1.600000e+006.261729e-16 4.551296e-01
17      1.700000e+006.349258e-16 4.551296e-01
18      1.800000e+006.429605e-16 4.551296e-01
```

19	1.900000e+006.504282e-16 4.551296e-01
20	2.000000e+006.574183e-16 4.551296e-01
21	2.100000e+006.639815e-16 4.551296e-01
22	2.200000e+006.701459e-16 4.551296e-01

7.4 Errorbar gm/Id vs. Square Law Equation



7.5 Heatmap W/L in Abhängigkeit von ft



8 Literaturverzeichnis

Https://www.studysmarter.de/. (2025). https://www.studysmarter.de/schule/physik/kernphysik/atomkern/

Matthes, E. (2023). *Python crash course: A hands-on, project-based introduction to programming* (3rd edition). No Starch Press.

McKinney, W. (2022). *Python for Data Analysis* (3. Aufl.). O'Reilly Media, Incorporated.

Oehme, W. F., Huemer, M., Pfaff, M., & Milosiu, H. (2025). *Elektronik und Schaltungstechnik. Ein verständlicher Einstieg.* (3. Aufl.).

Wilke, C. O. (2025). *Fundamentals of Data Visualization*. https://clauswilke.com/dataviz/index.html