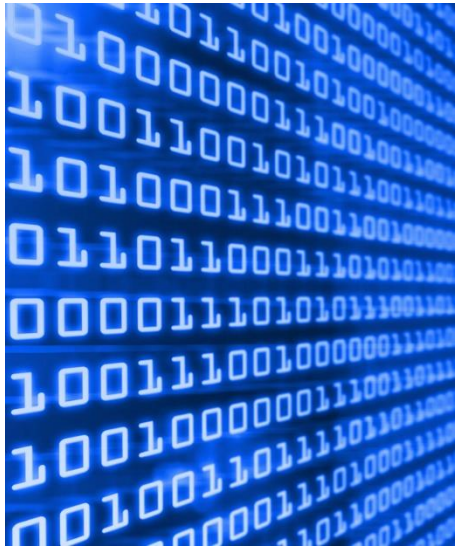
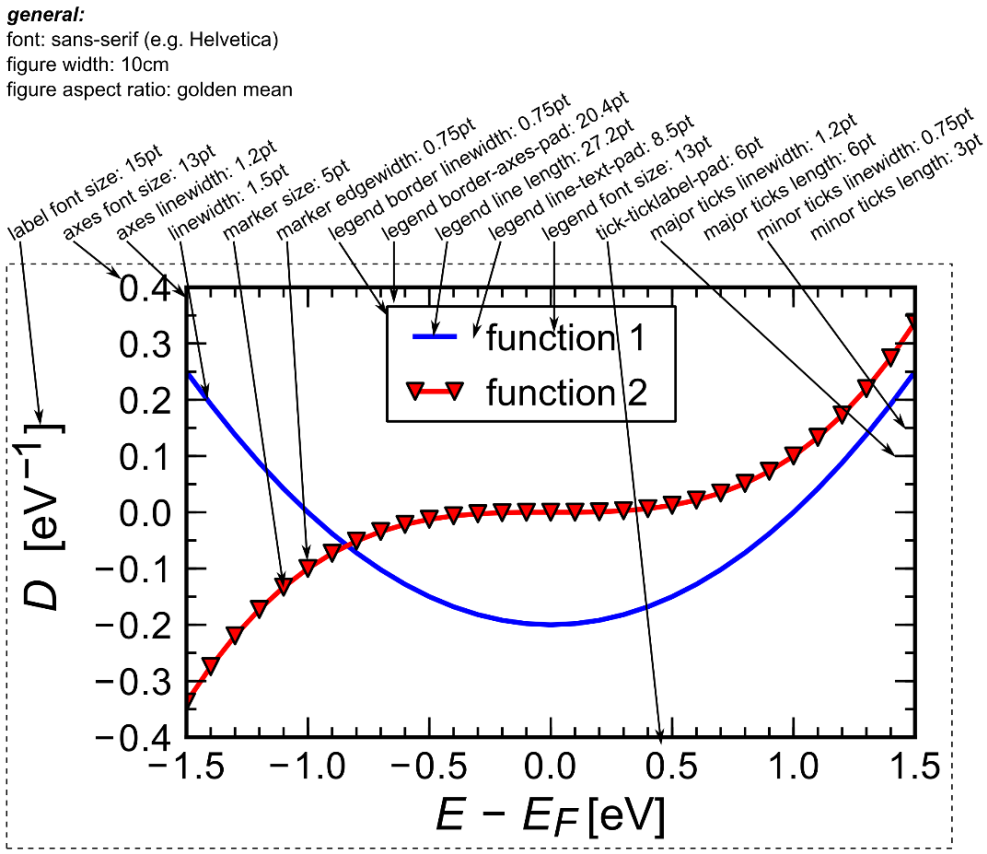
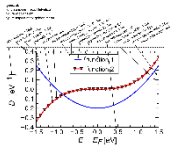
Programming Core Skills - vom „output“ zur Abbildung

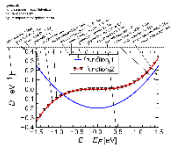






Einführung – warum eigentlich als Thema?

- Effizientes Arbeiten / Gute Gewohnheiten, um wissenschaftliche Abbildungen zu erstellen
  - Werkzeuge und Arbeitsfluss zur Generierung von Grafiken
  - Verwendungsformen wissenschaftlicher Grafiken
  - Grundlegende Arten von Grafiken
  - Notizen zur farblichen Gestaltung



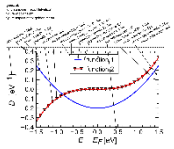
Werkzeuge

Beispiele hier

Übungen

- Python/Matplotlib (http://www.matplotlib.org/)
- Gnuplot (http://www.gnuplotting.org/ - gute Seite mit VIELEN Tipps für Gnuplot)
- Matlab (https://de.mathworks.com/products/matlab.html) / GnuOctave / FreeMat)
- Origin (http://originlab.com) / SciDaVis (http://scidavis.sourceforge.net)
- Mathematica (http://www.wolfram.com/mathematica/)
- Latex / pgfplots (http://pgfplots.sourceforge.net)
- Excel / OpenOffice?

- **Grundsatzfrage: Zieldokument in MS Office / OpenOffice oder Latex?**
  - **Abhängig von „Institutskultur“ etc.**



Excel / OpenOffice / LibreOffice?

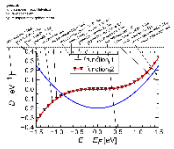
Entgegen (historisch gut begründeter Vorurteile) ist seit MS Office 2013 das Erstellen wissenschaftlich ausgereifter Grafiken in Excel möglich. Allerdings mit einiger Handarbeit.

Vorteile:

- Daten immer "fertig" im HG, Wechsel des Stils möglich
- WYSIWYG Editor; kleine Einstiegshürde
- Gute Weiterverwendung -> Anbindung an Powerpoint / Publisher / Word

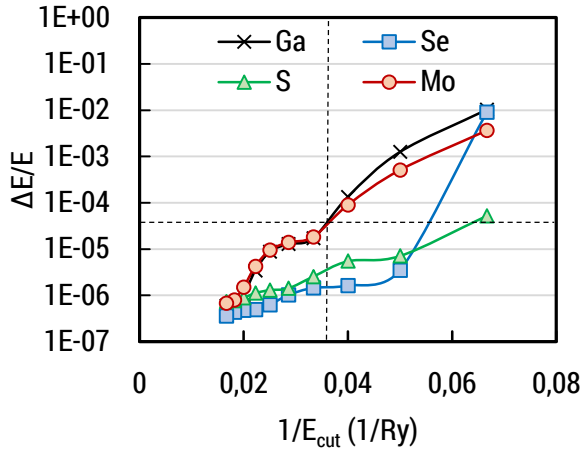
Nachteile:

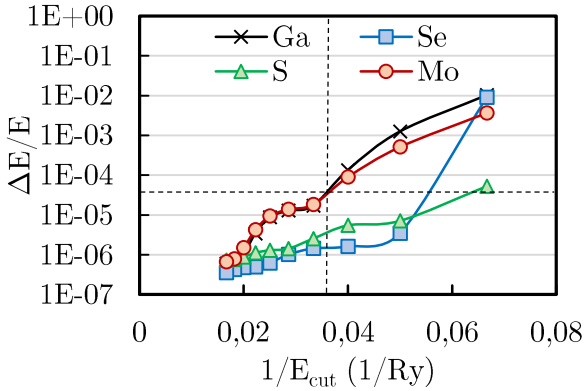
- Weniger Analysemöglichkeiten als in wissenschaftlicher Software (fitten)
- Händisches Importieren von Daten
- Beschriftungen etc. unflexibel (Gleichungen/Formelzeichen …)
- Templates oft sperrig – letztlich viel Handarbeit; aber gute Nachnutzung
- Excel: Kommerzielle Software / Kopplung an Win/Mac (kein Linux)
- Kompatibilität mit LibreOffice (open source, auch Linux) begrenzt



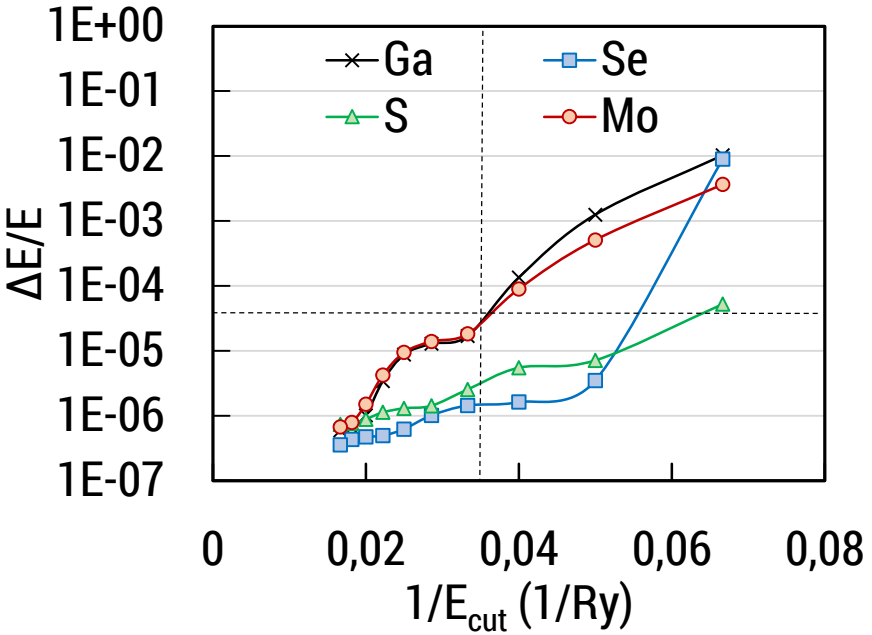
Beispiele in Excel

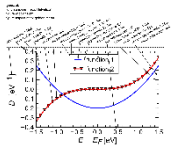
Aufbereitet für Textdokument / 2-spaltig





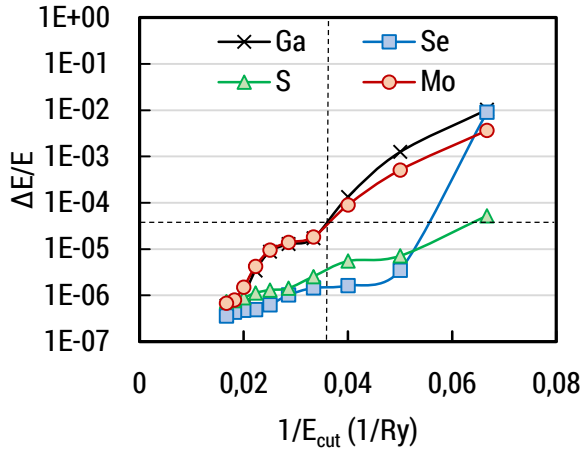
Aufbereitet für Präsentation (nur in ppt!)

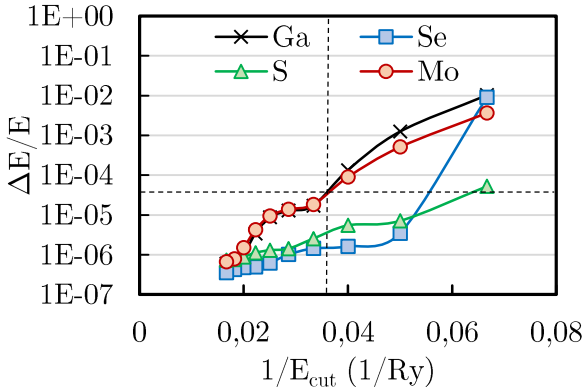




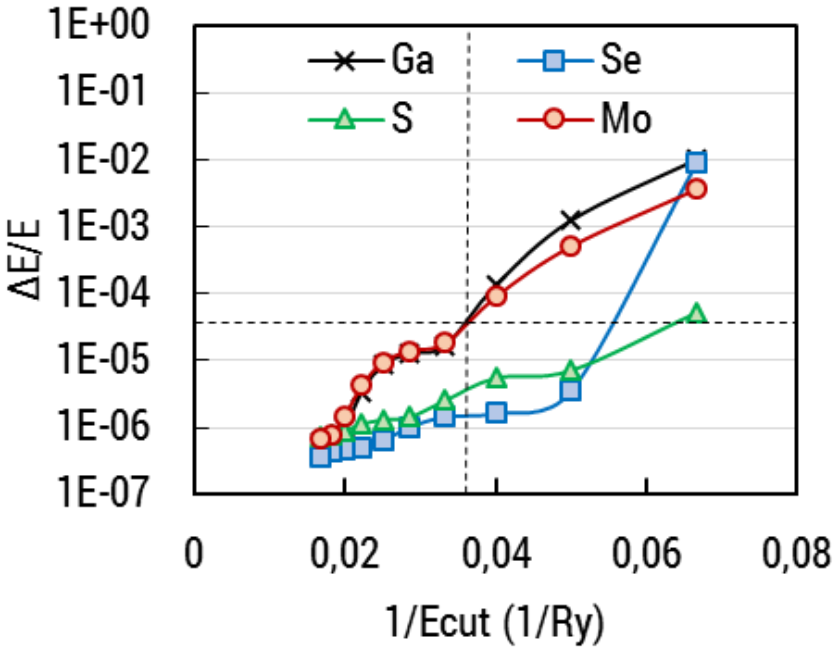
Beispiele in Excel

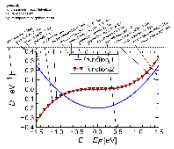
Aufbereitet für Textdokument / 2-spaltig





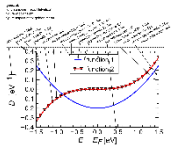
Aufbereitet für Präsentation (nur in ppt!)





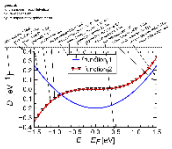
Verwendungsformen wissenschaftlicher Abbildungen

- Textpublikation (A4: 21 x 29.7cm)
  - Schriftart serif oder serifenlos;
    mind. 10pt (für Mikrobeschriftung mind. 8 pt)
  - Breite ca. 7.5-8 cm (halbseitig); ca. 15-16 cm (vollseitig)
- Präsentation / Folien (Folie ca. 25x20 cm)
  - Schriftart serifenlos; mind. 16 pt (14pt Mikrobeschriftung)
  - Ca. 10-12 cm Breite (halbseitig)
- Posterpräsentation (A0, ca. 89 x 120 cm)
  - Schriftart serifenlos; mind. 24 pt (20pt Mikrobeschriftung)
  - Ca. 20cm Breite; Meist gute Folien/Textgrafiken geeignet



Workflow: Gute Praxis

- Daten auswerten und in Textdatei geben (ggfs. komprimiert, falls viele Daten)
  - Plotten: Rohdaten aus Textdatei holen
- Nachteil:
  - Anfangs Mehraufwand
- Vorteil:
  - Auswertung und Plotten voneinander getrennt (Fehleranfälligkeit von Skripten)
  - Neu plotten geht schnell – Anpassung des Darstellungsstils
  - Gute Nachnutzung in vielen Programmen
  - Erzeugung mehrerer Stile (Poster / Folien / Text) gleichzeitig möglich

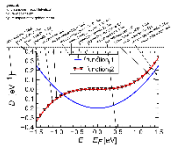


Workflow: Gute Praxis

- Workflow abhängig vom Ziel der Darstellung

Anzahl erzeugter Grafiken

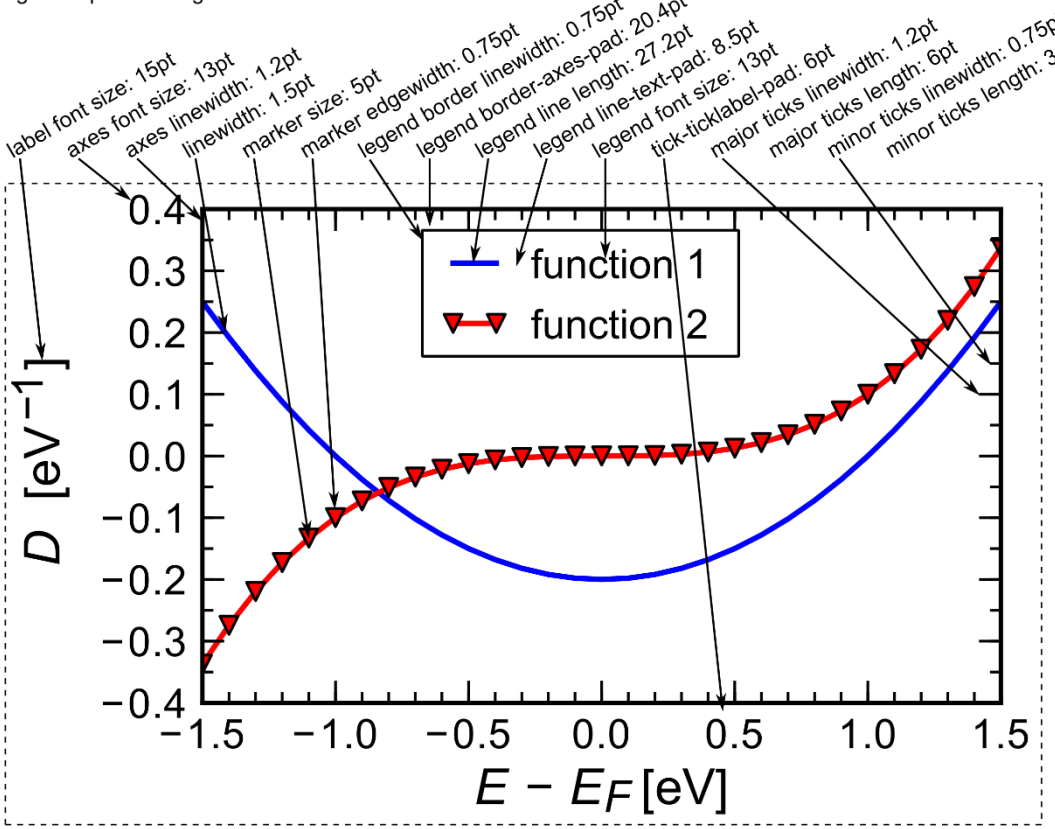
- Produktionsreife Publikationsgrafik / Präsentationsgrafik
  - Was will ich mit der Grafik aussagen?
    - NUR das Nötigste! (Ockham‘s Rasiermesser)
  - Workflow oben absolut sinnvoll – inkl. Feinschliff
  - Stil konsistent halten (möglichst nur EIN Werkzeug verwenden!)
- Interne Präsentation
  - Wenigstens einfach lesbare Achsenbeschriftung und wissenschaftlich klar – wenn auch nicht ausgefeilt
    - Workflow oben möglichst einhalten; kein Feinschliff
- „Schnell mal angucken“
  - Dann so **hässlich** wie möglich, um der Versuchung vorzubeugen, die Abbildung weiter zu nutzen!

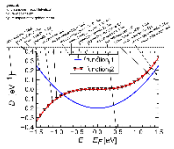


Warum möglichst ein Werkzeug für publikationsreife Grafiken?

- „Aus einem Guss“ (Wissenschaftlichkeit auch in der Form)
- Man muss sehr, sehr viel spezifizieren, wenn man das Werkzeug wechselt…

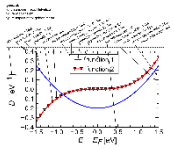
general:
font: sans-serif (e.g. Helvetica)
figure width: 10cm
figure aspect ratio: golden mean





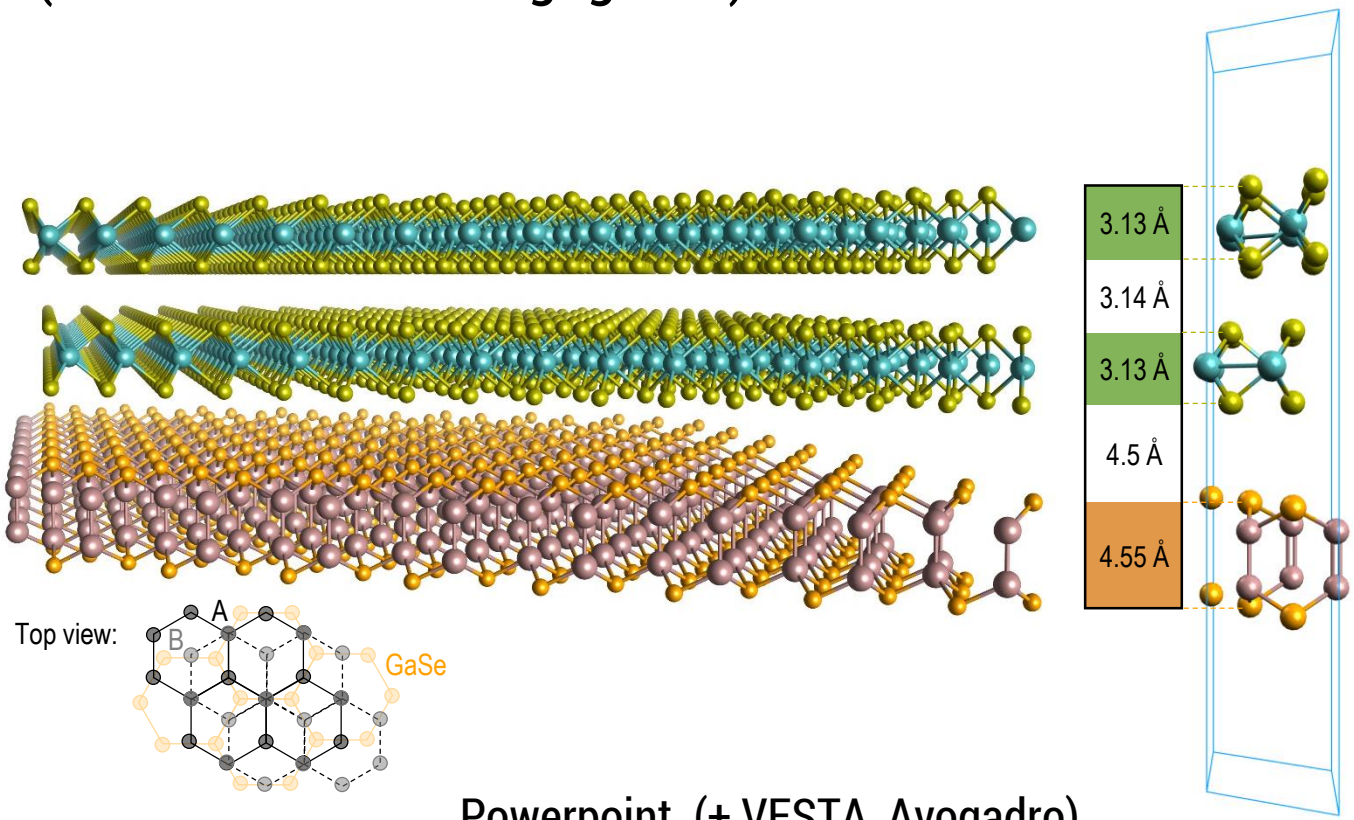
Workflow: Gute Praxis

- Händische Nacharbeit (Inkscape, Powerpoint)?
  - Vermeiden, wo immer es geht!
  - Nur (!) bei Grafiken für die wissenschaftliche Publikation oder Präsentation (am bestem im Präsentationswerkzeug)
  - Nicht für Entwurf oder interne Zwecke
  - Nur dann, wenn Umsetzung im Skript sehr aufwändig ist UND klar ist, dass das Ergebnis passt!
    - Allerallerletzter Feinschliff!
- Warum?
  - Große Zeitinvestition – v.a. bei Korrektur kleiner Errata
- Besser:
  - Textboxen / Untergrafiken dann doch per Skript…

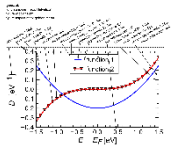


Workflow: Gute Praxis

- Beispielgrafik für händische Nacharbeit (Reine Visualisierungsgrafik)

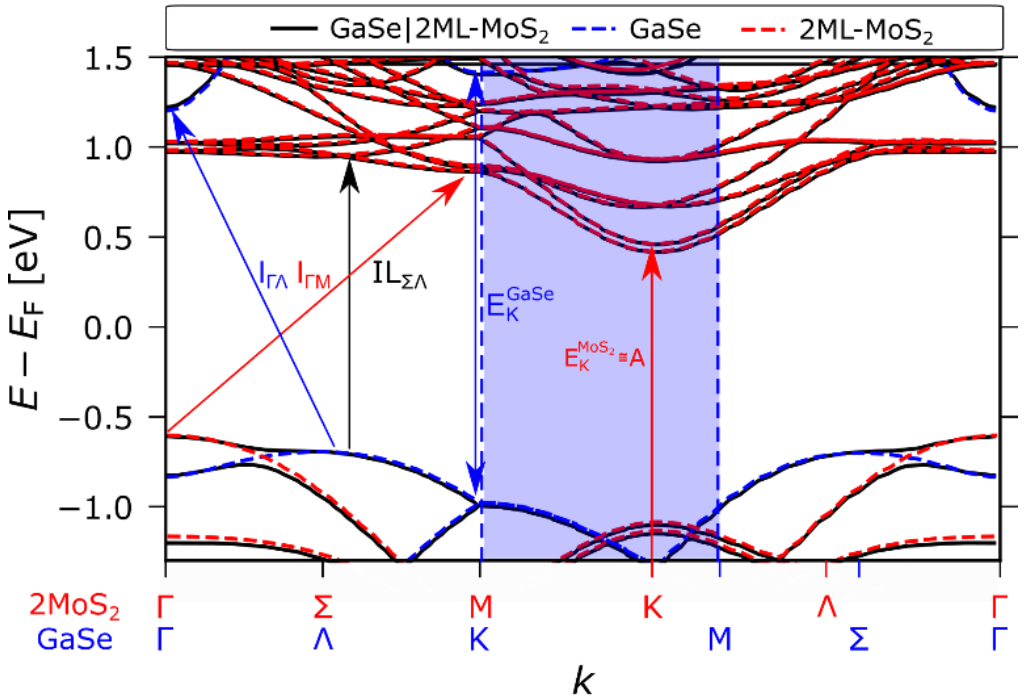


Powerpoint (+ VESTA, Avogadro)

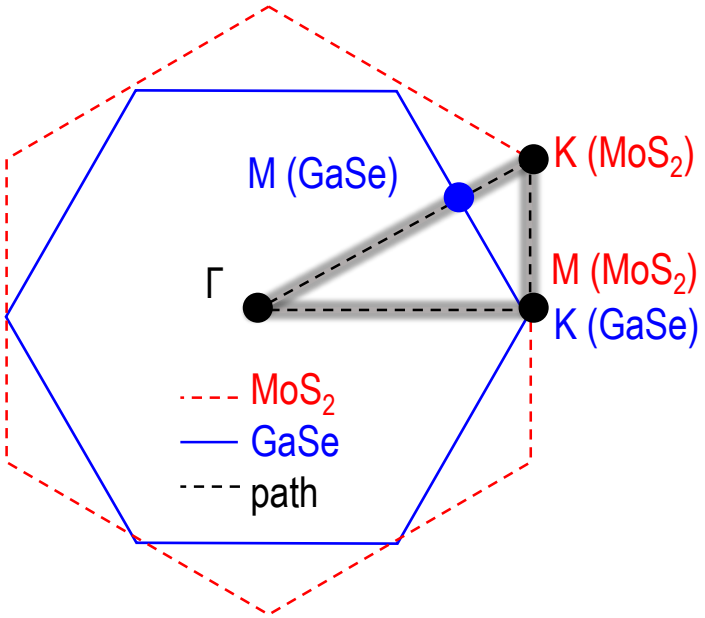


Workflow: Gute Praxis

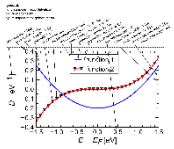
- Beispielgrafik für händische Nacharbeit (Publikationsgrafik, schematische Skizze)



Inkscape

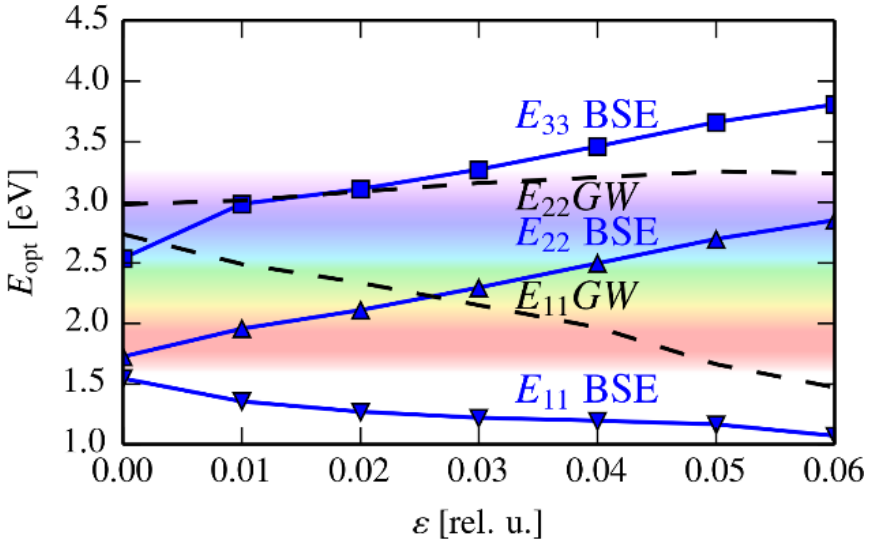


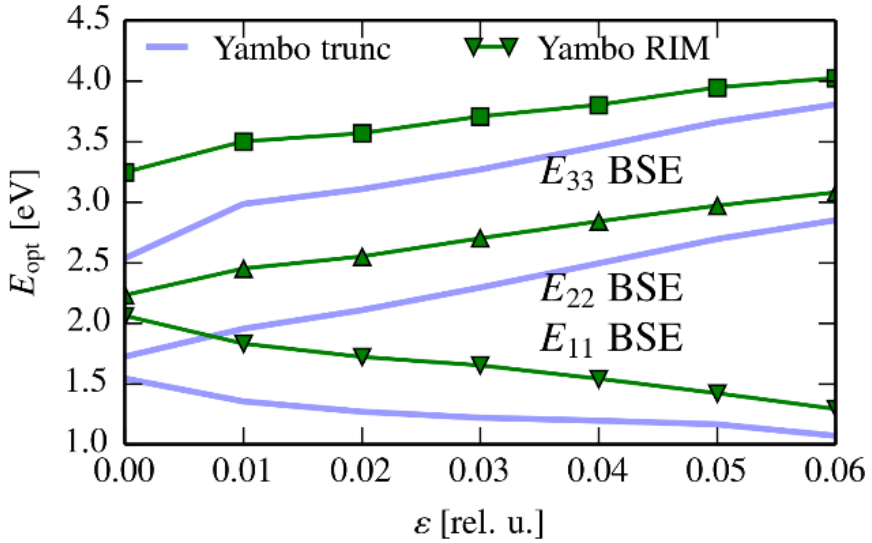
Powerpoint



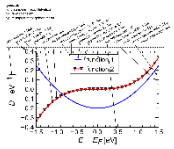
Workflow: Gute Praxis

- Beispielgrafik ohne händische Nacharbeit (Publikationsgrafik, einspaltig; Vergleichbarkeit!)



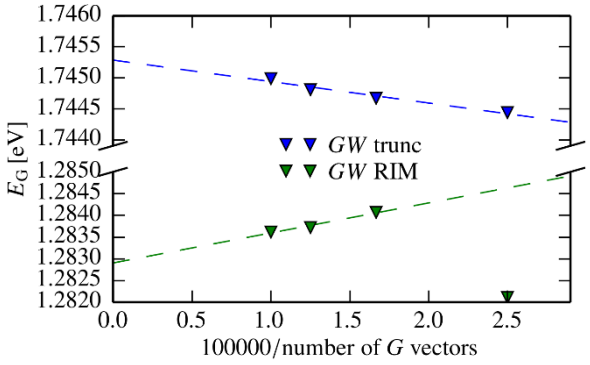


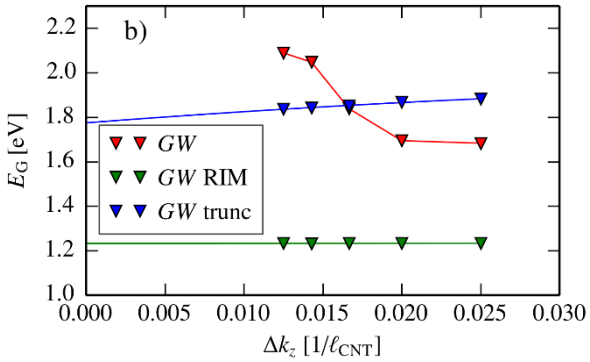
Python / matplotlib

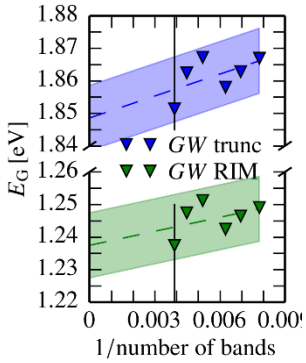


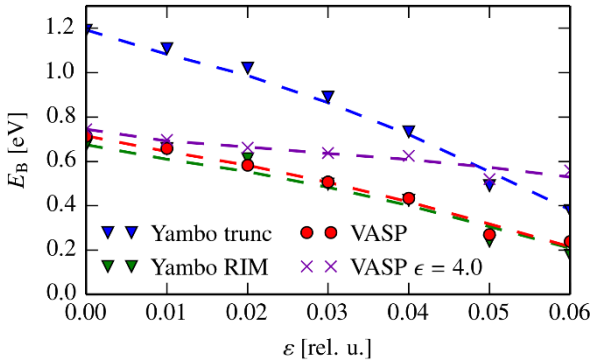
Workflow: Gute Praxis

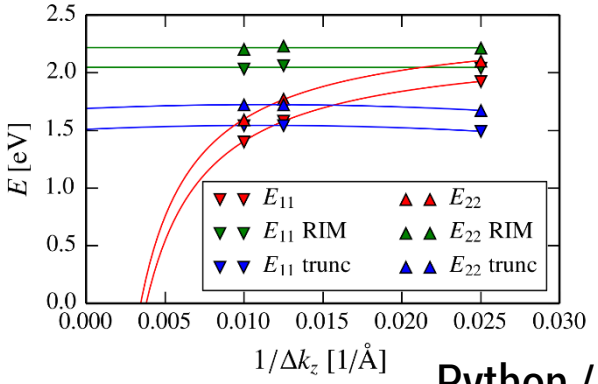
- Beispielgrafik ohne händische Nacharbeit (Publikationsgrafiken; Konvergenzstudie Theorievergleich; Vergleichbarkeit!)

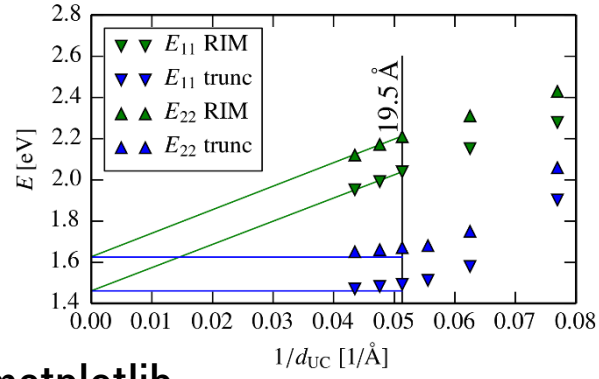




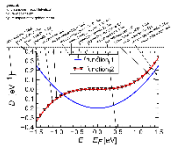




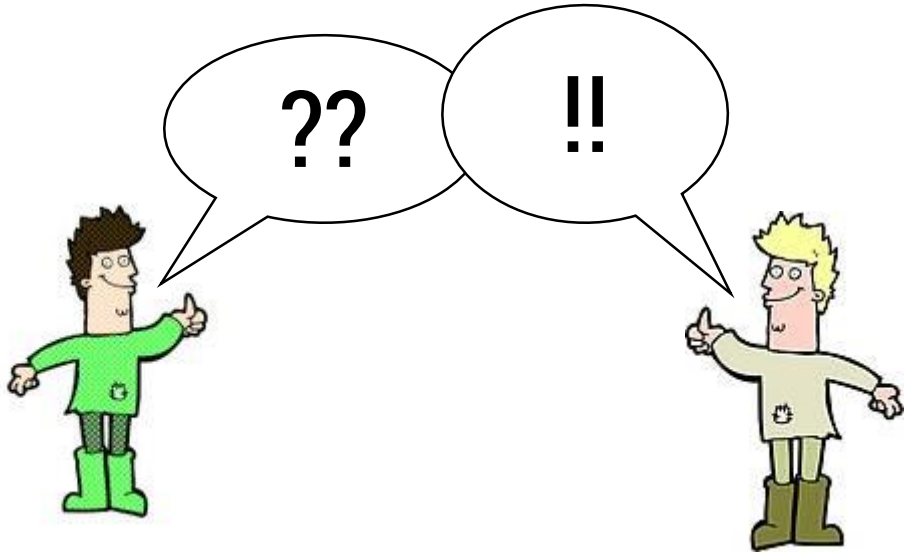


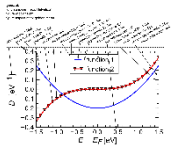


Python / matplotlib

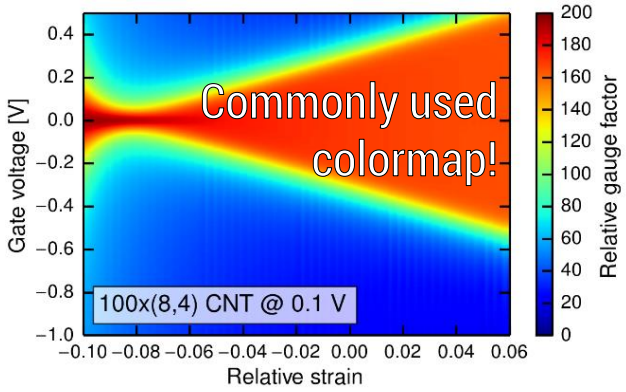


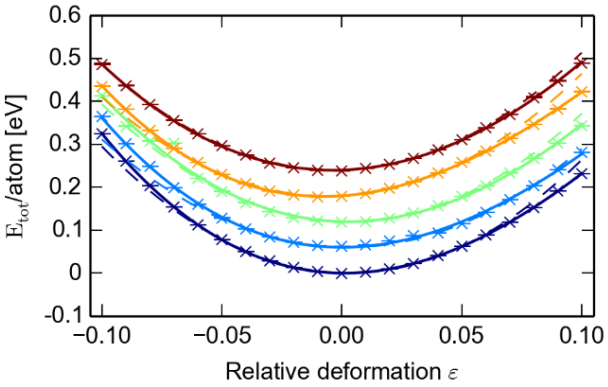
Fragen oder Diskussion zum Arbeitsfluss / guter Praxis / Verwendungsformen?





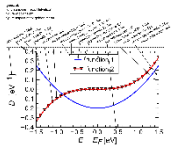
Colormaps



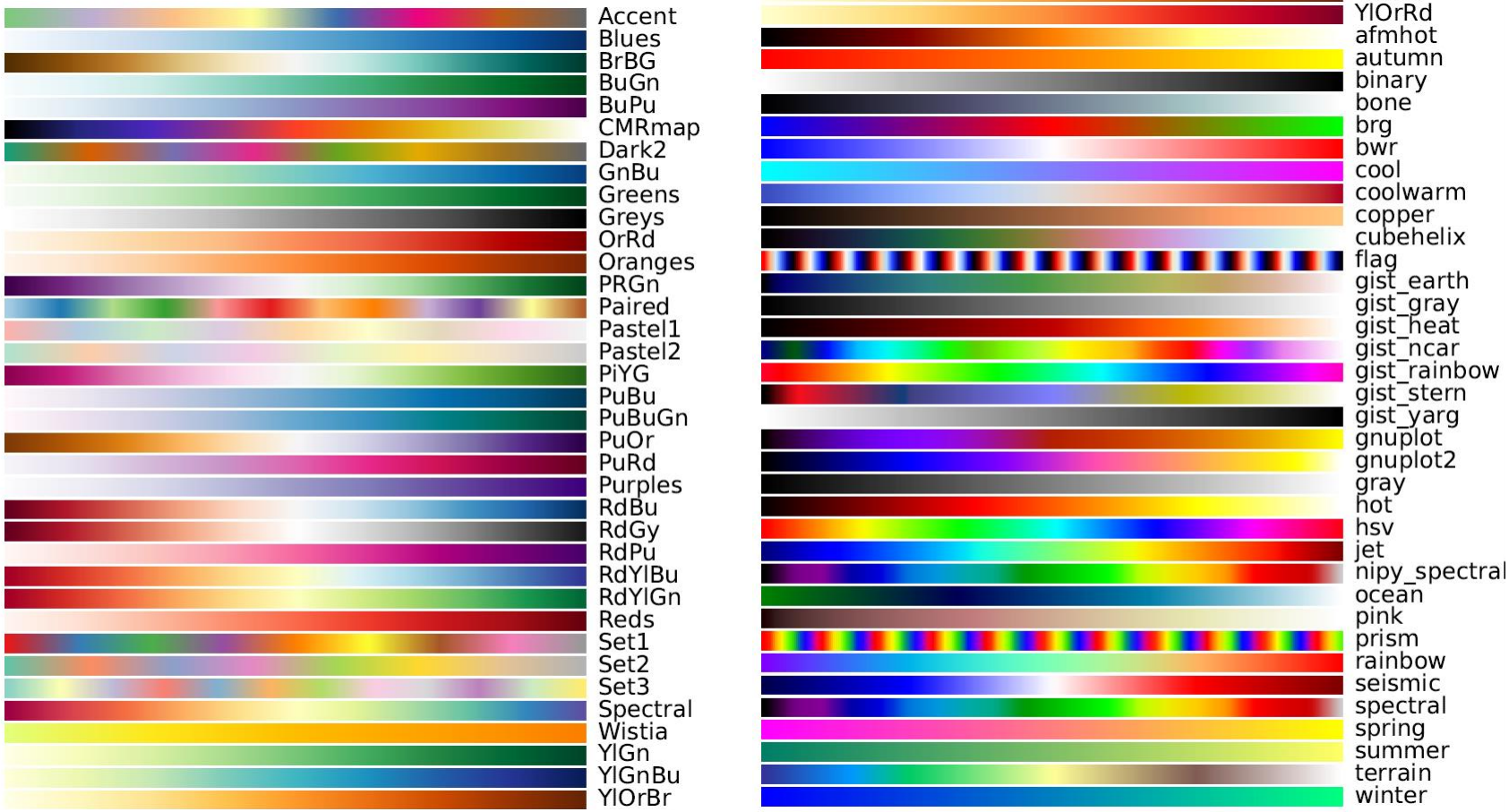


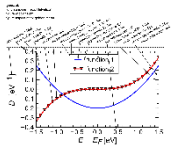
- Precise display of data (line plots & 2d plots)
- Colorblind people / grayscale readability
- Red/green blind people
- Good representability on screen and ink

- Needs:
  - Good contrast (> 1 color)
  - Dark-to-bright colormap
  - Avoid red-green transitions (better: avoid both colors in one colormap)
  - Isometric (differences between colors scale with differences in data)

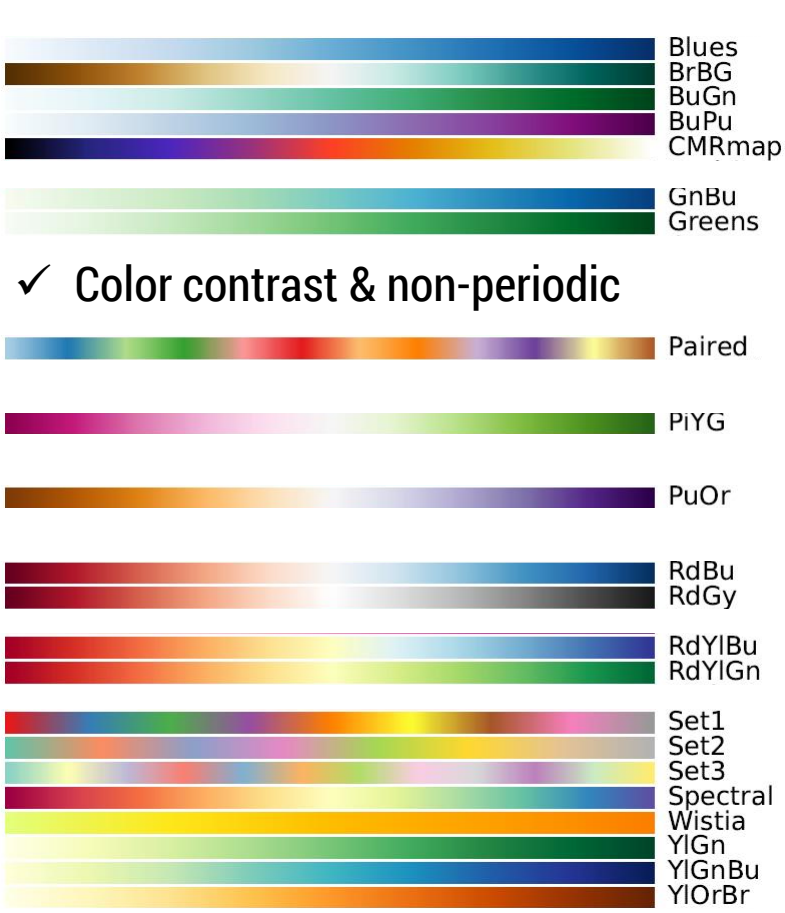


Available colormaps (python – matplotlib 2.0)

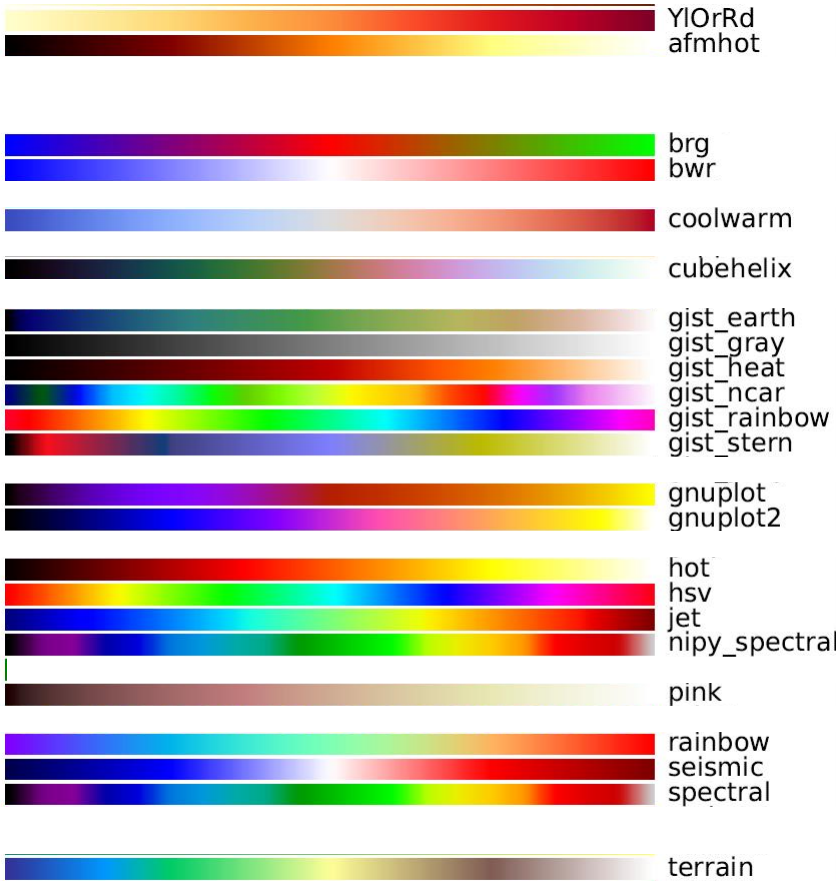


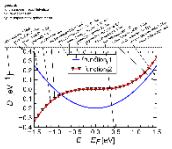


Available colormaps (python - matplotlib)

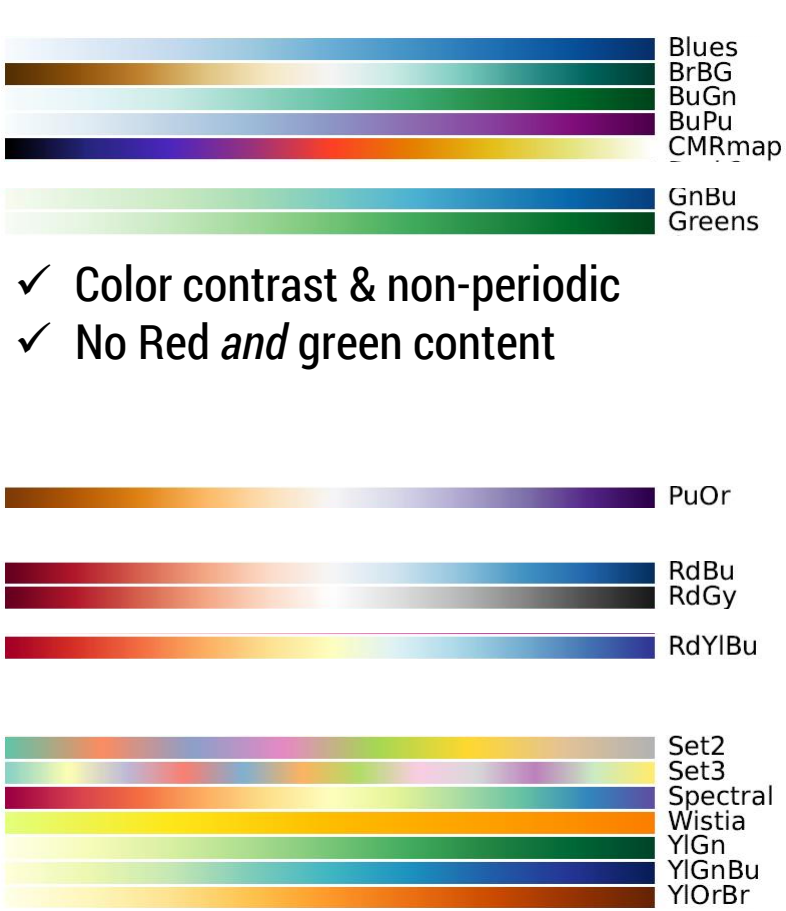


✓ Color contrast & non-periodic

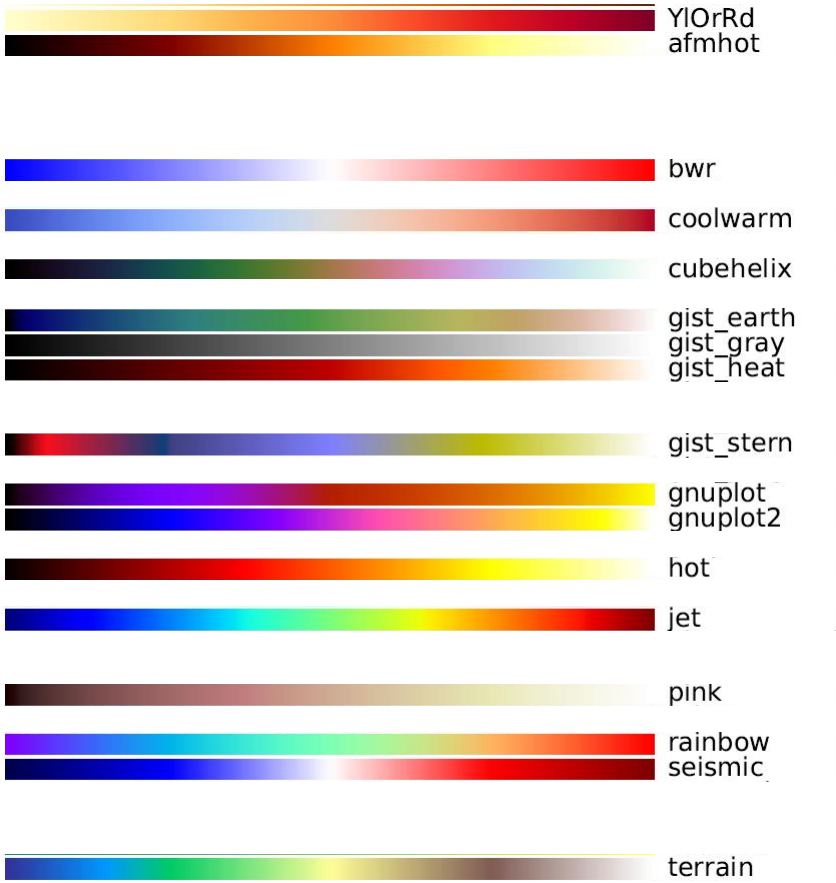


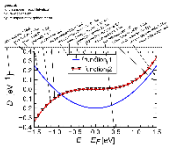


Available colormaps (python - matplotlib)

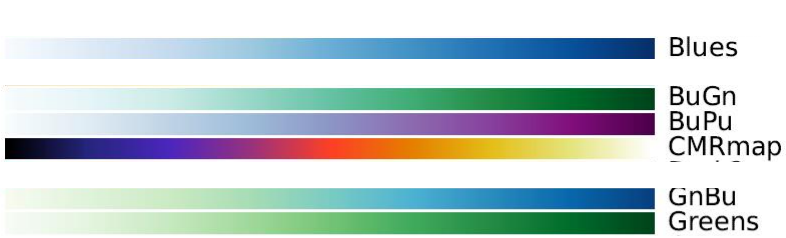


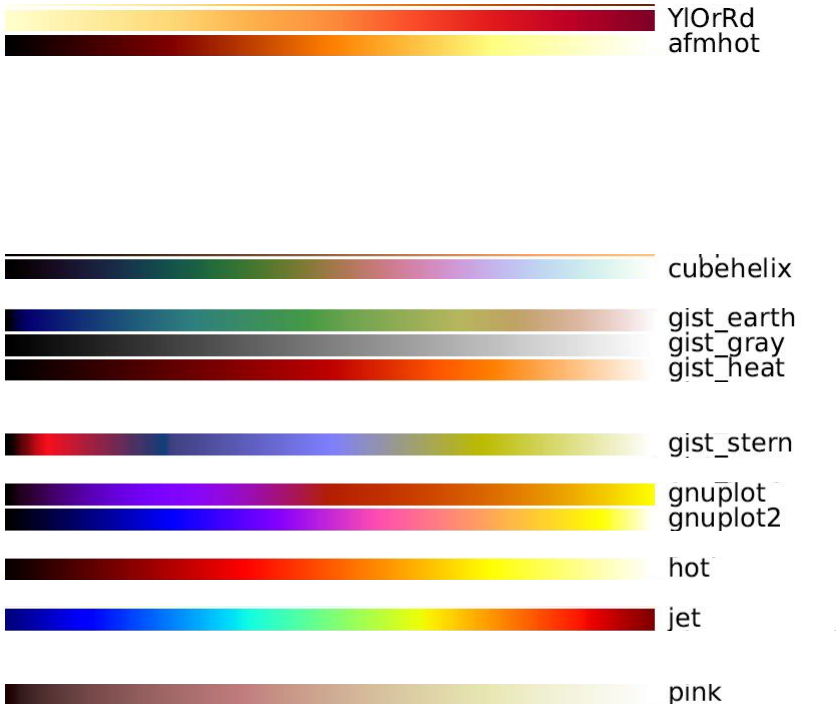
- ✓ Color contrast & non-periodic
- ✓ No Red *and* green content



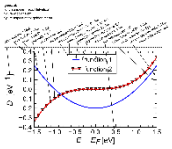


Available colormaps (python - matplotlib)



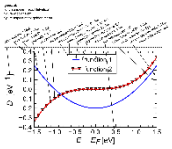


- ✓ Color contrast & non-periodic
- ✓ No Red *and* green content
- ✓ Dark to bright (sequential, not diverging)

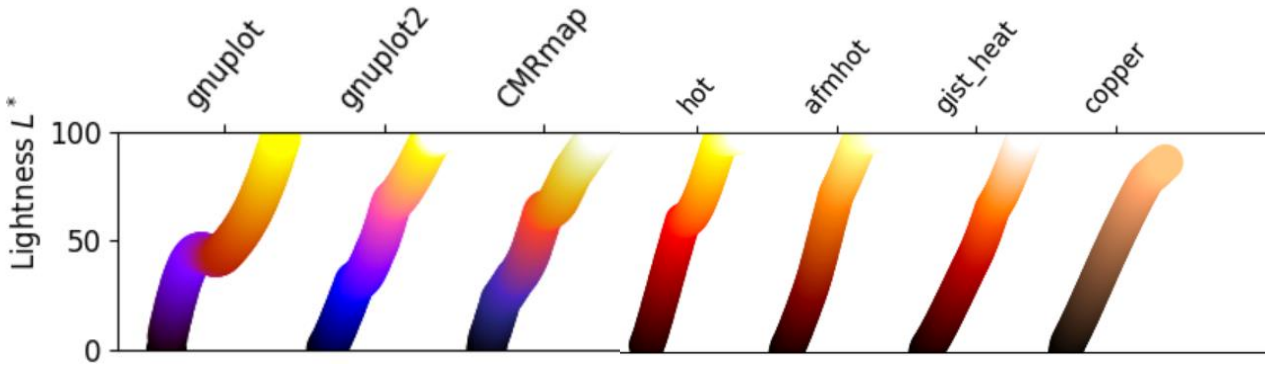


Available colormaps (python - matplotlib)

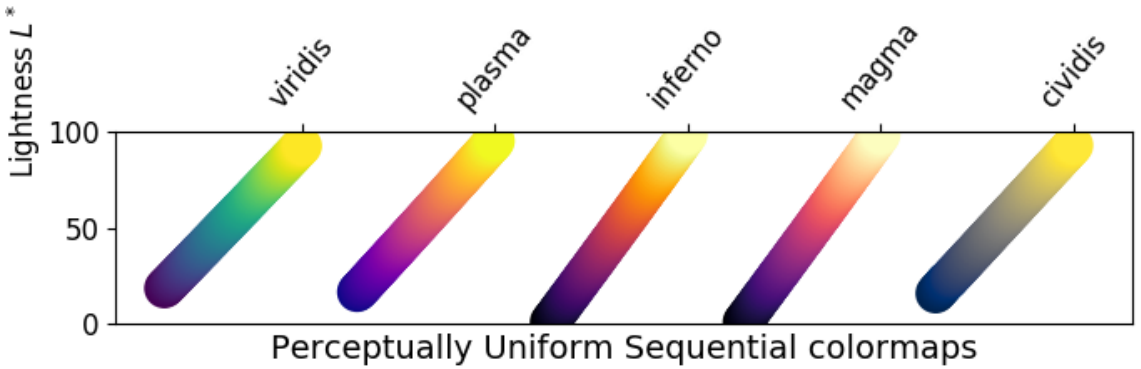
- ✓ Color contrast & non-periodic
- ✓ No Red *and* green content
- ✓ Dark to bright (sequential, not diverging)
- ✓ No white



Available colormaps (python - matplotlib)

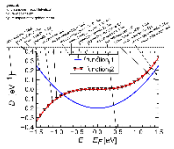


- ✓ Color contrast & non-periodic
- ✓ No Red *and* green content
- ✓ Dark to bright (sequential, not diverging)
- ✓ No white
- ✓ Perceptually uniform*

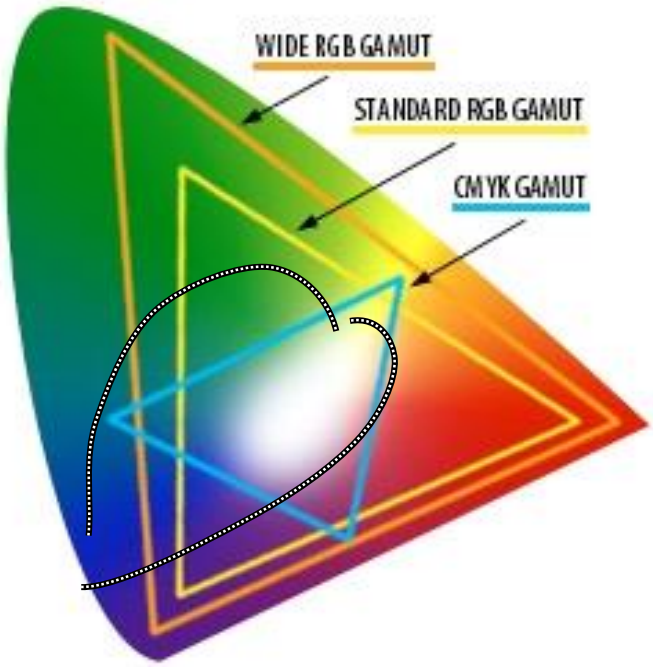


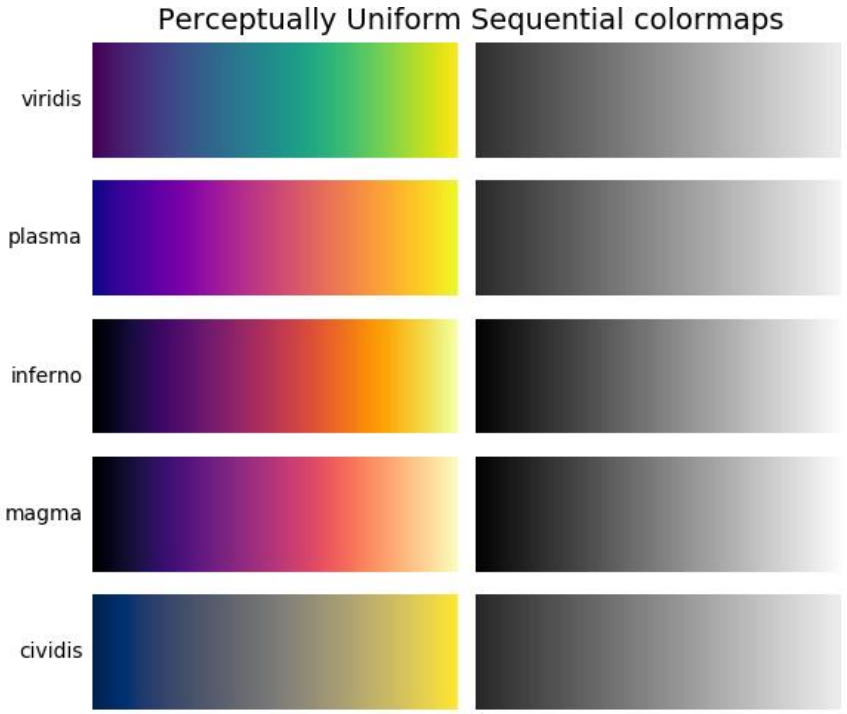
*and friendly for any color perception disease

https://matplotlib.org/users/colormaps.html, 28.11.2019



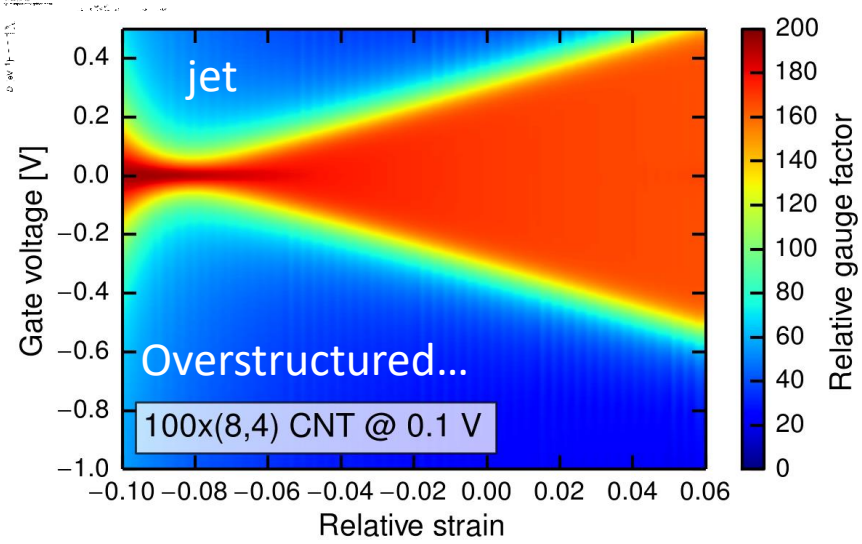
Available colormaps

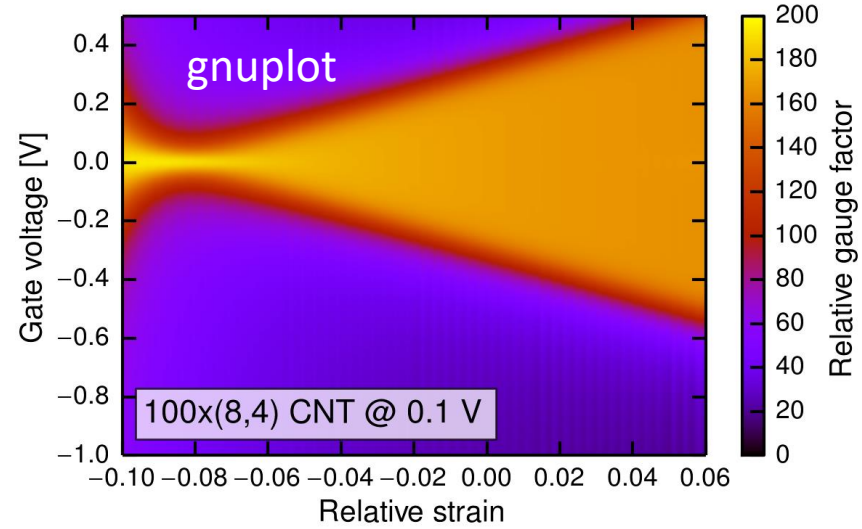




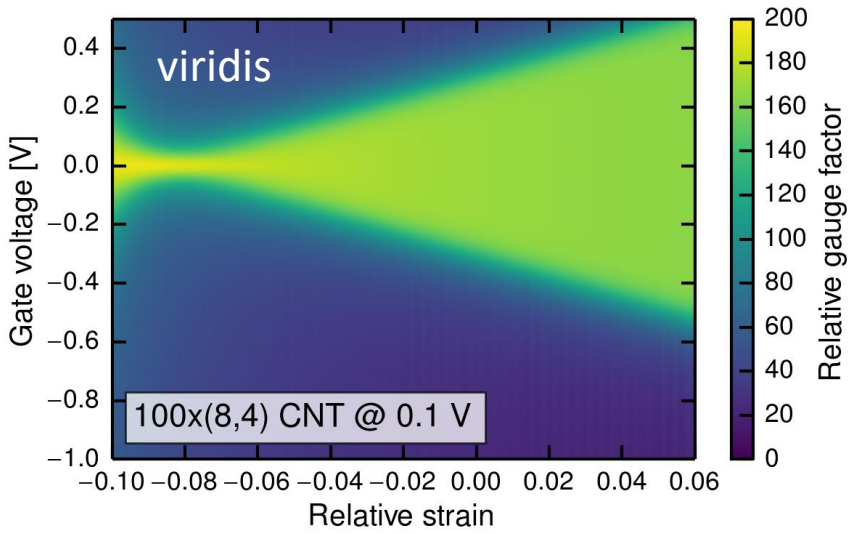
Warme Empfehlung: https://matplotlib.org/users/colormaps.html
Mehr „perceptually uniform maps“ unter https://colorcet.pyviz.org/
Mehr Hintergründe: https://arxiv.org/abs/1712.01662

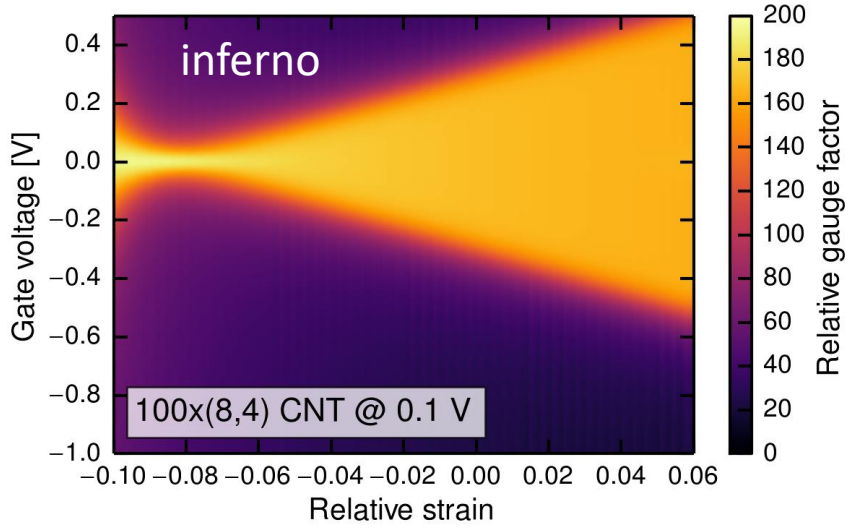
Conventional



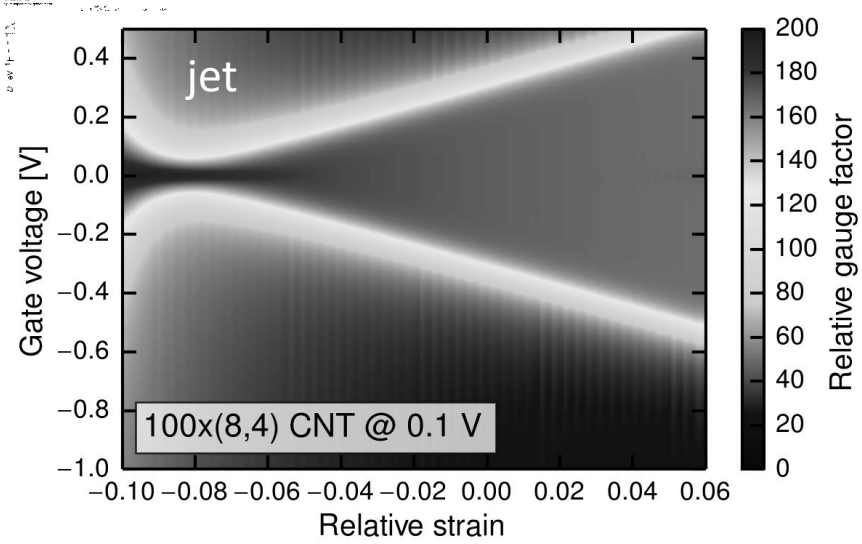


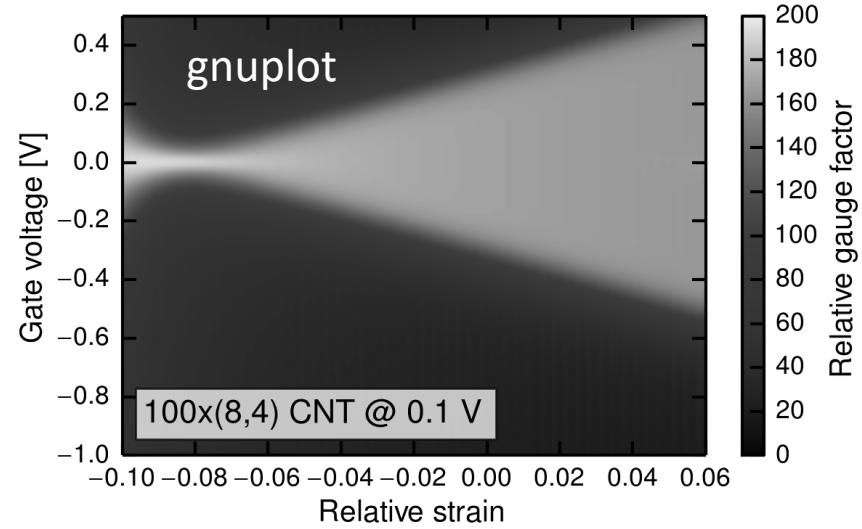
Novel



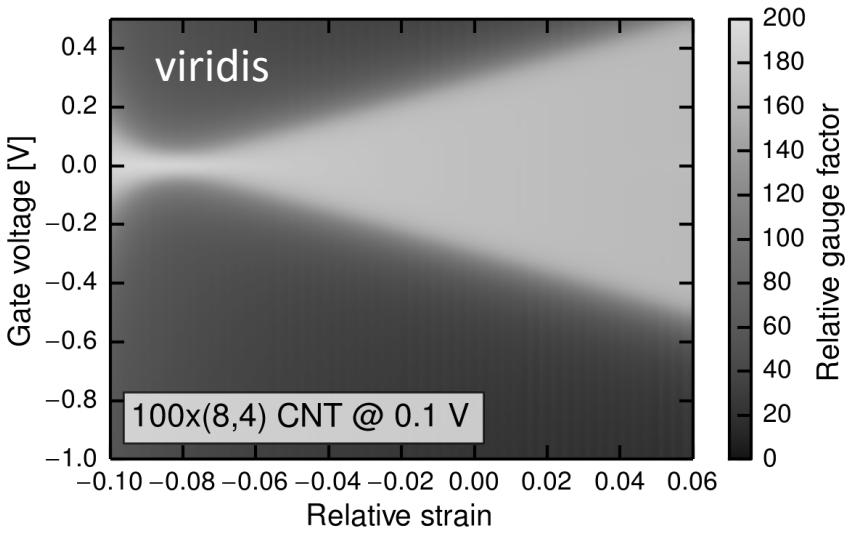


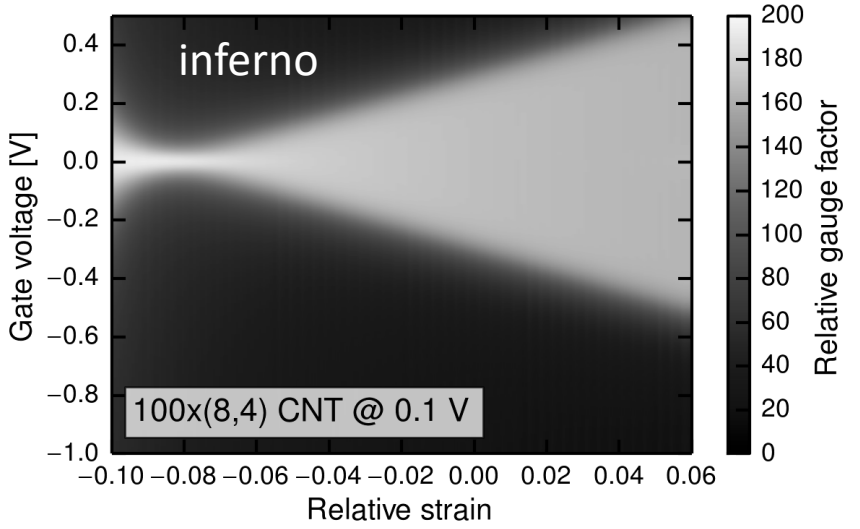
Conventional



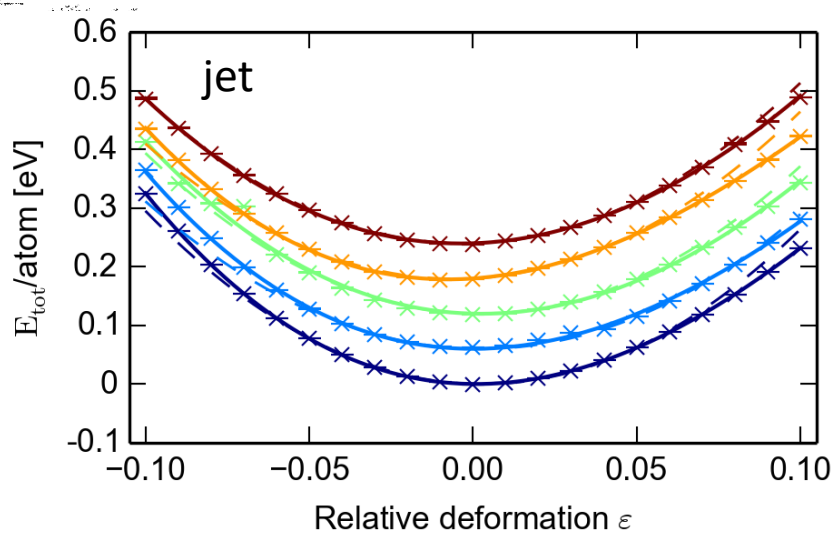


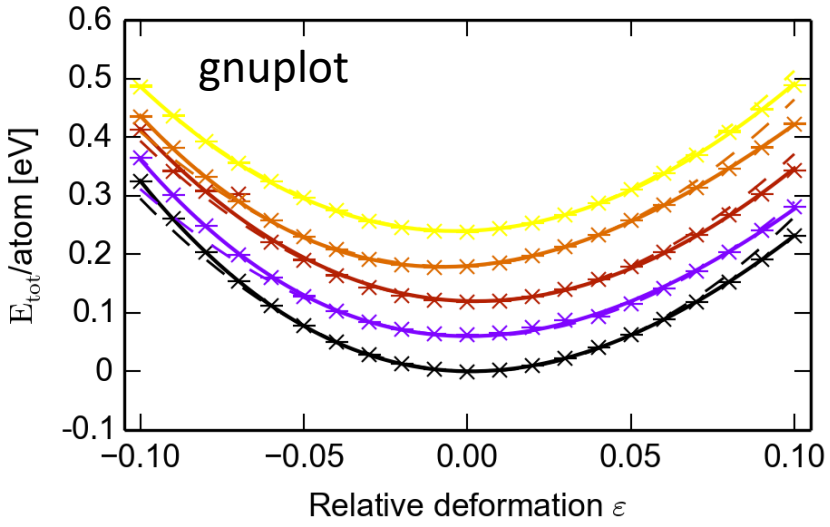
Novel



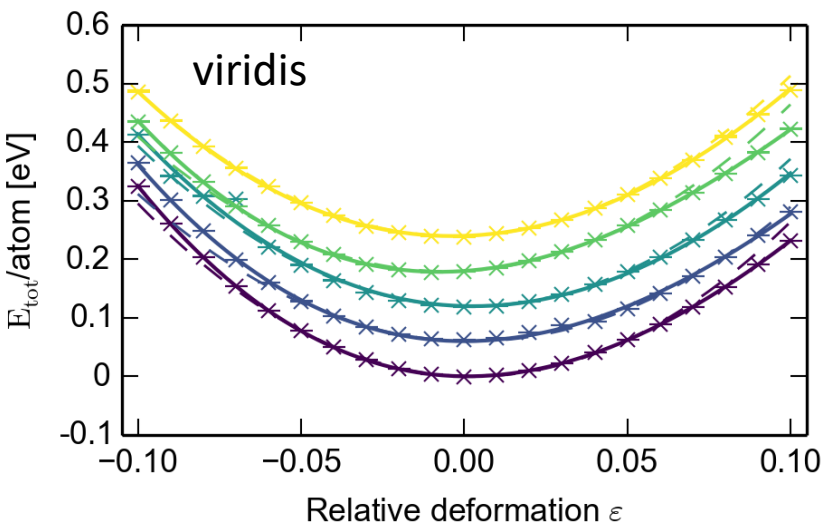


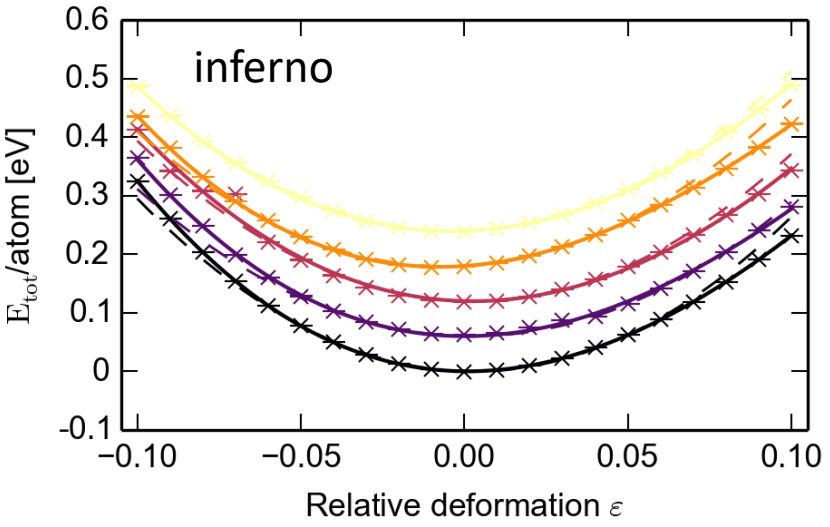
Conventional



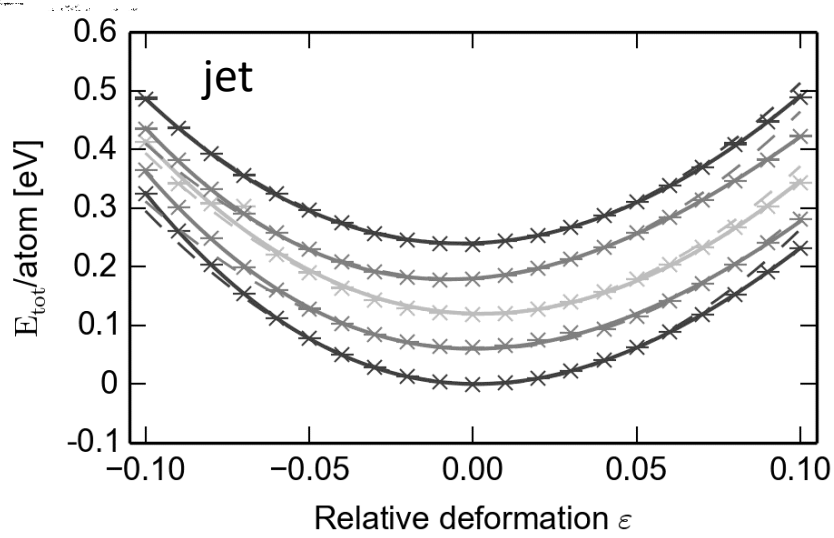


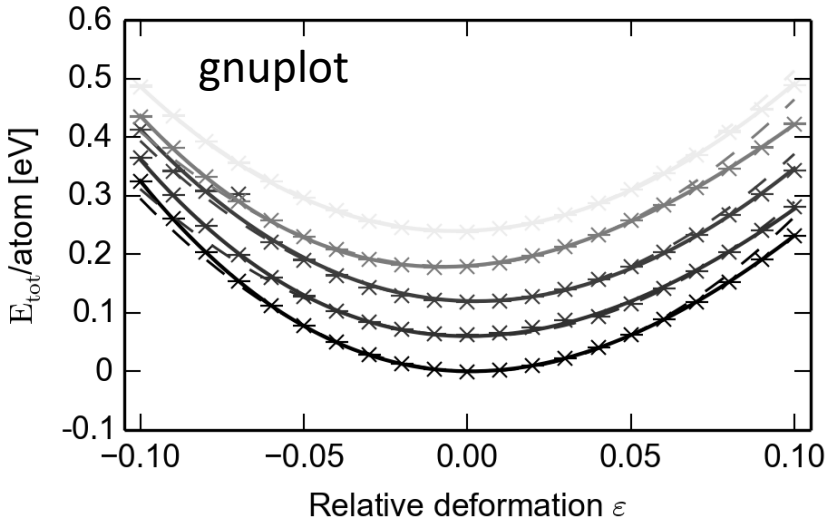
Novel



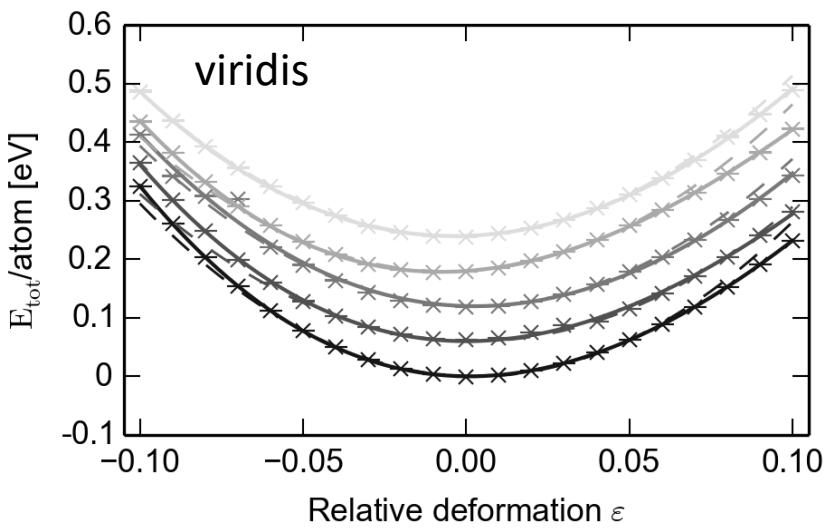


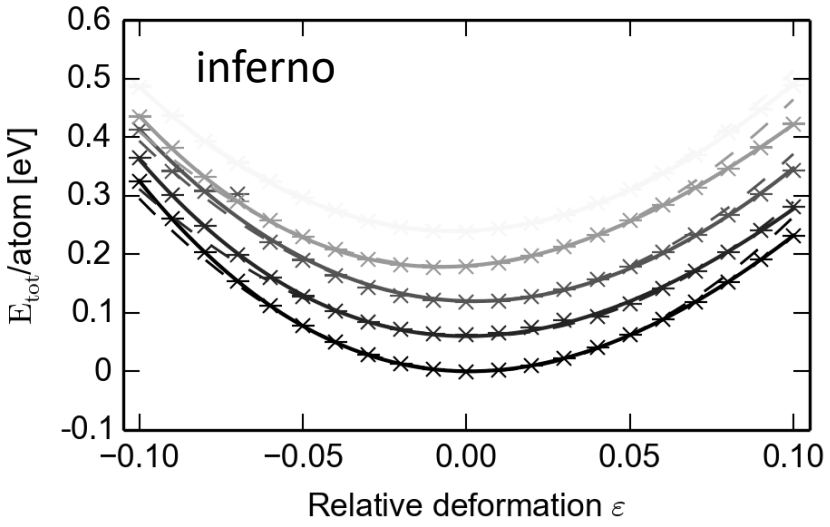
Conventional



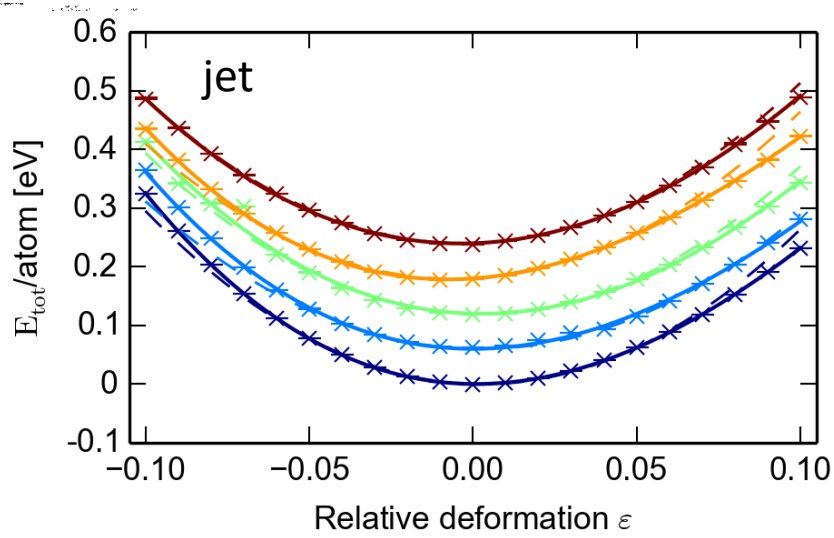


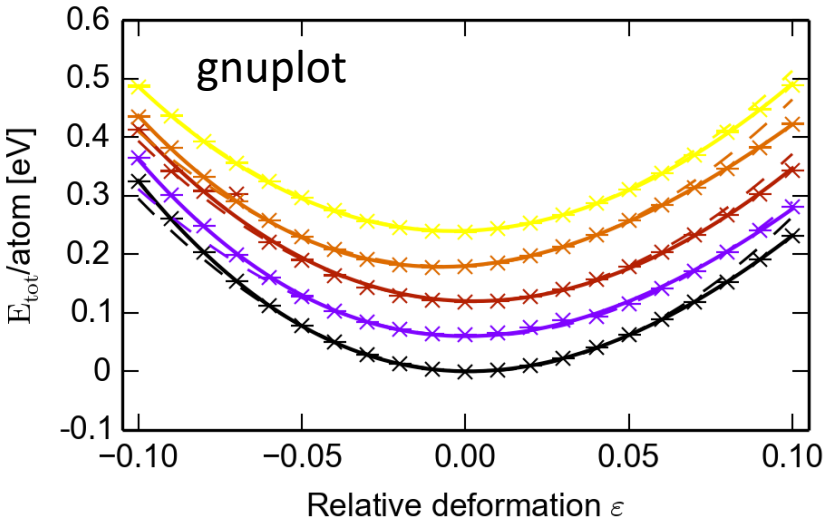
Novel



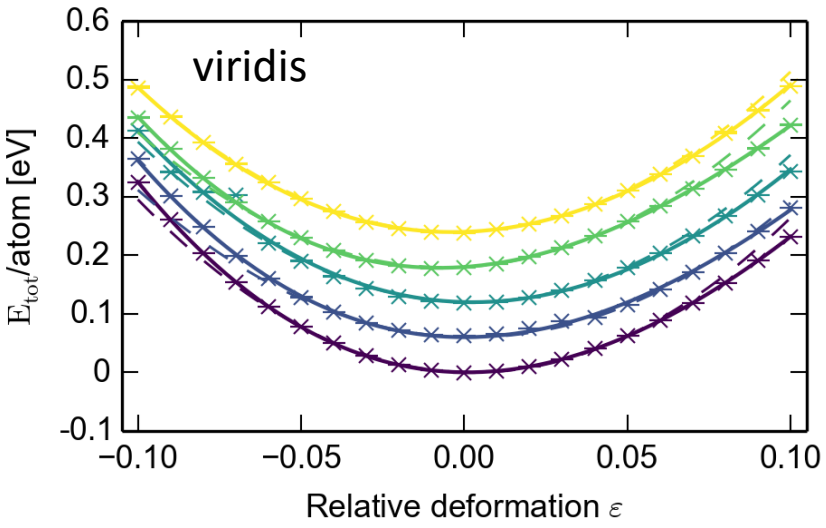


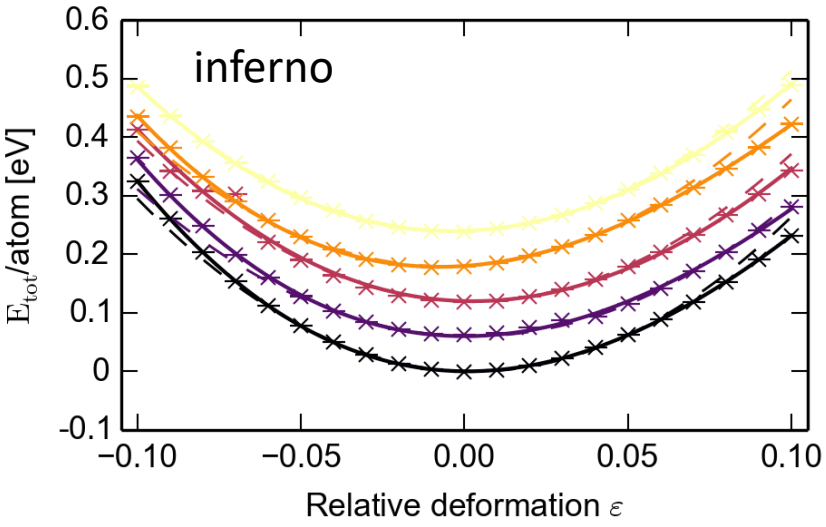
Conventional



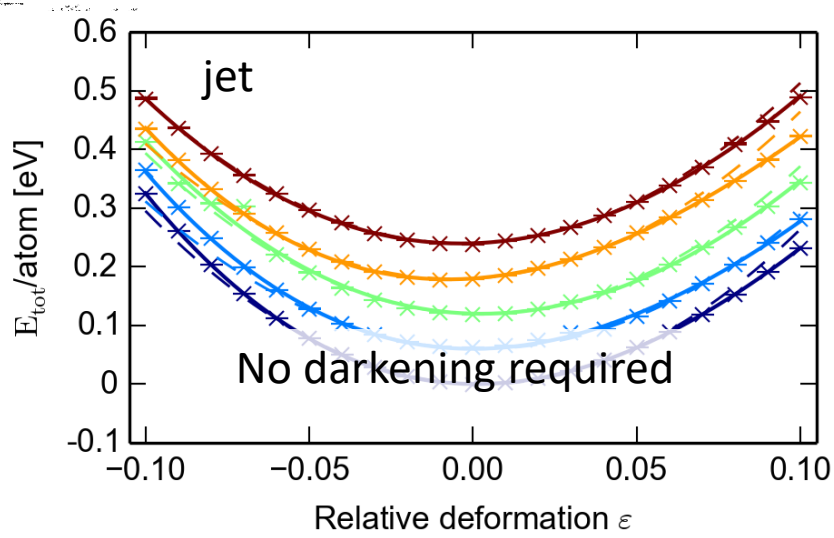


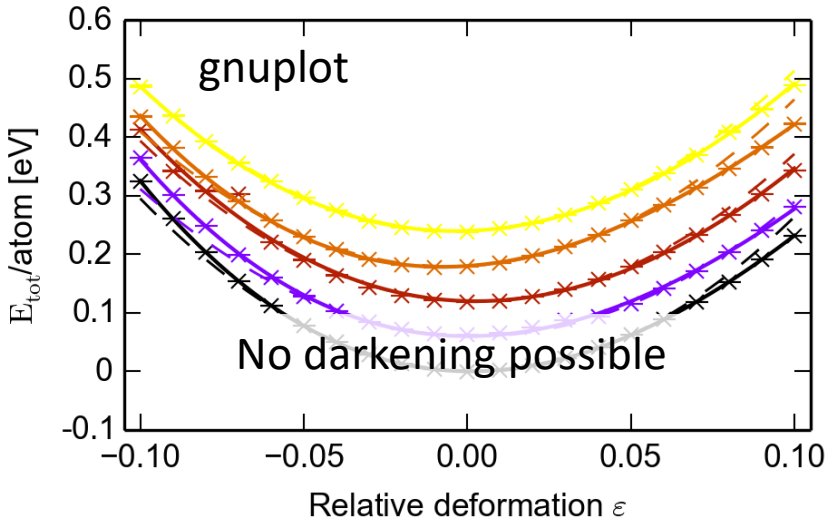
Novel



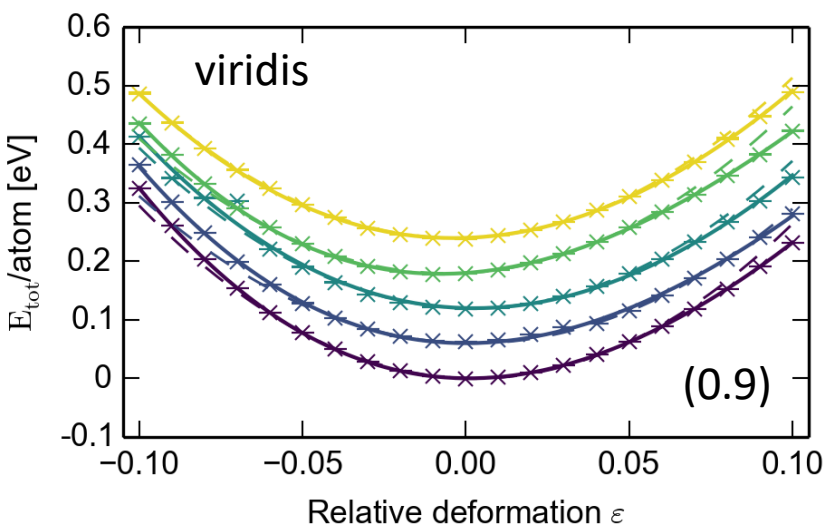


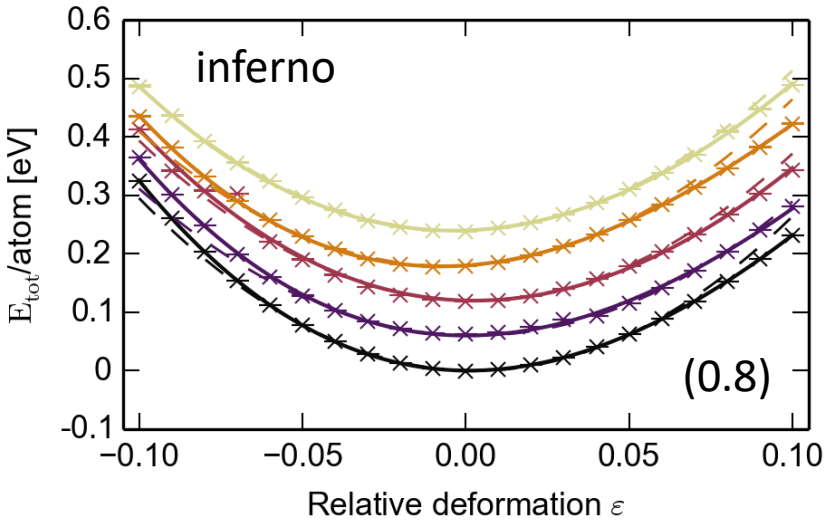
Conventional



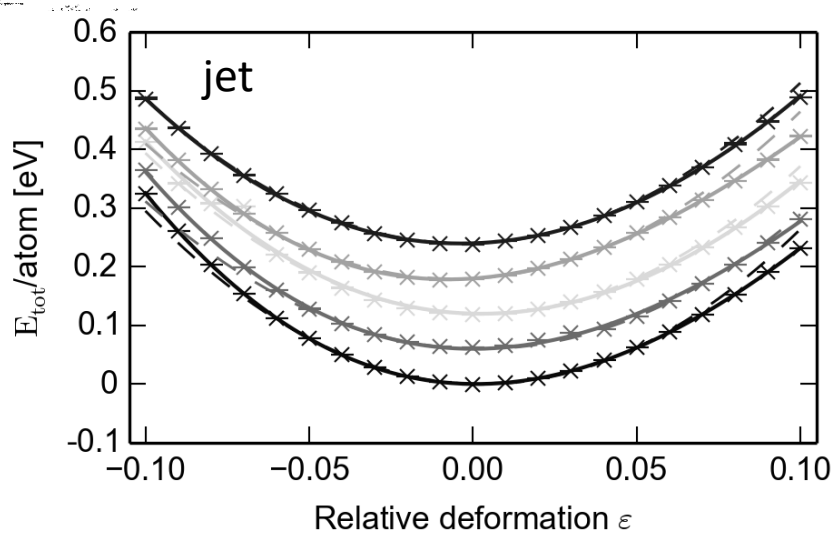


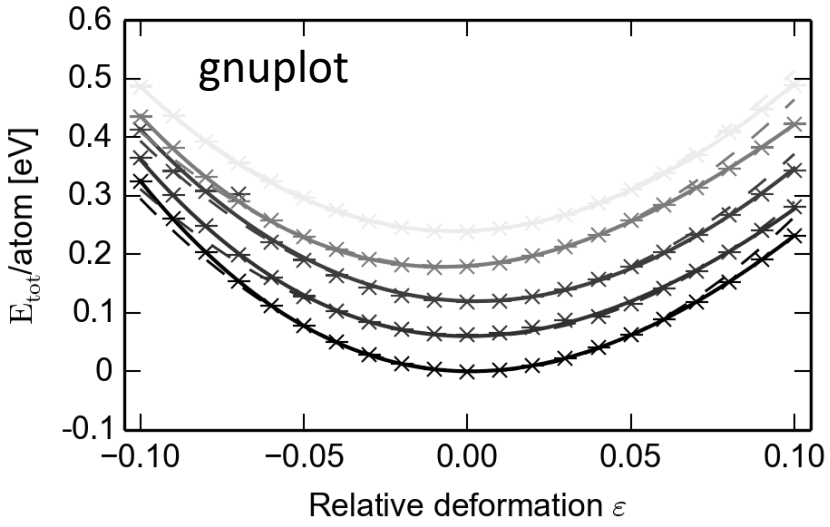
Novel



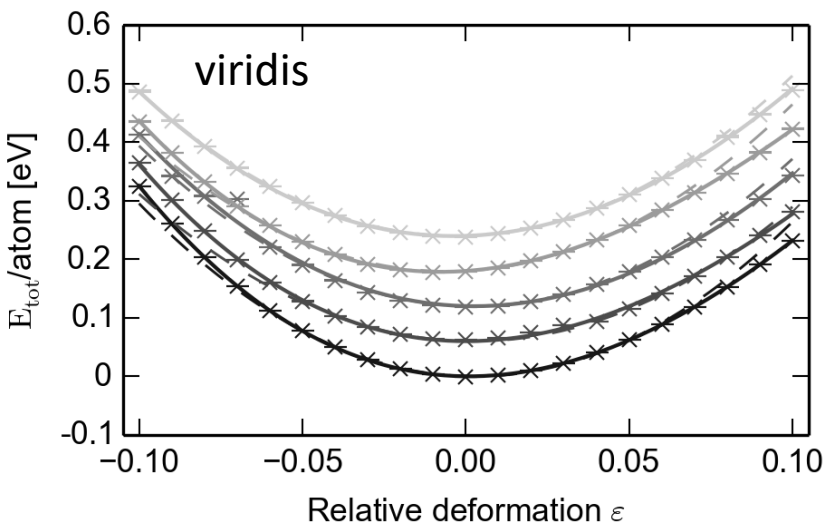


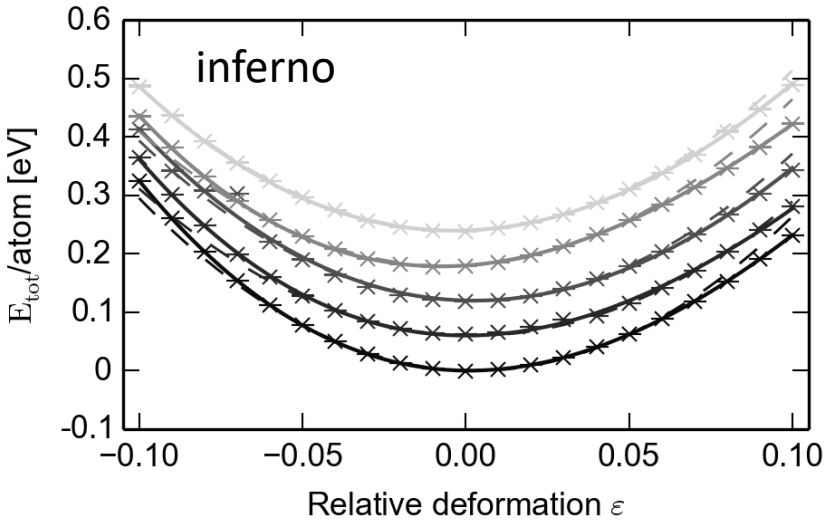
Conventional

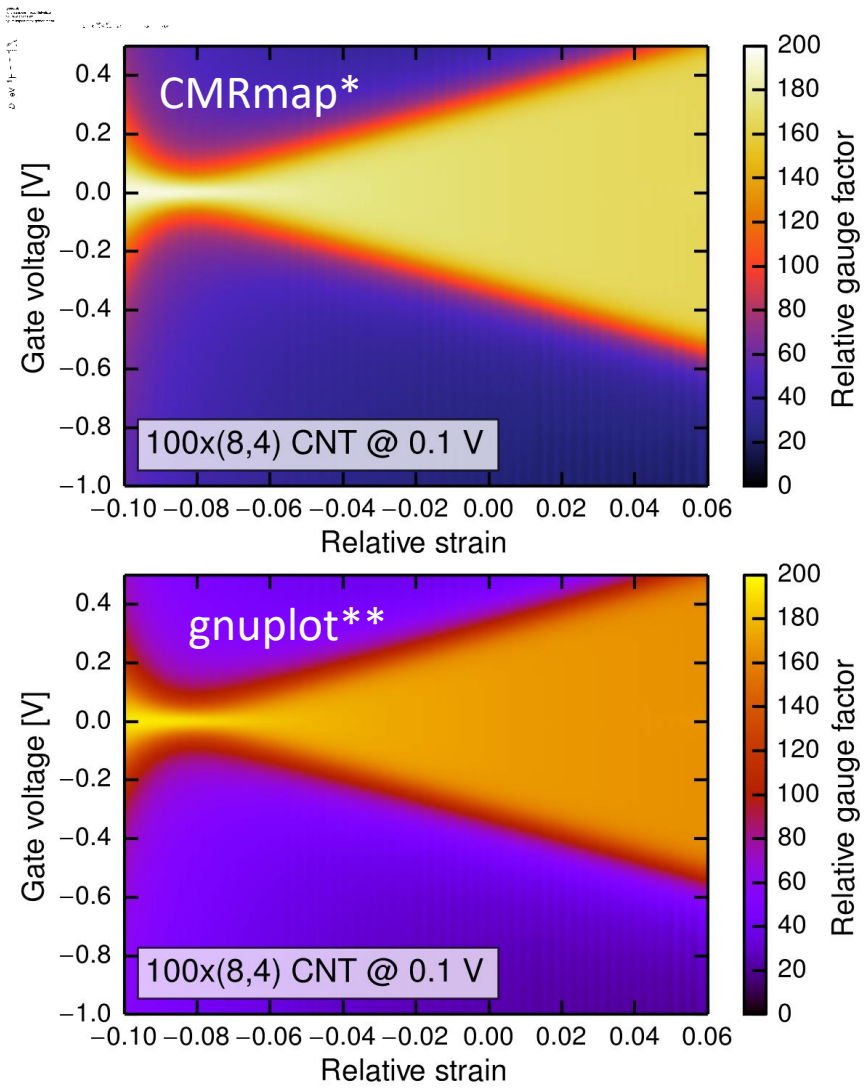


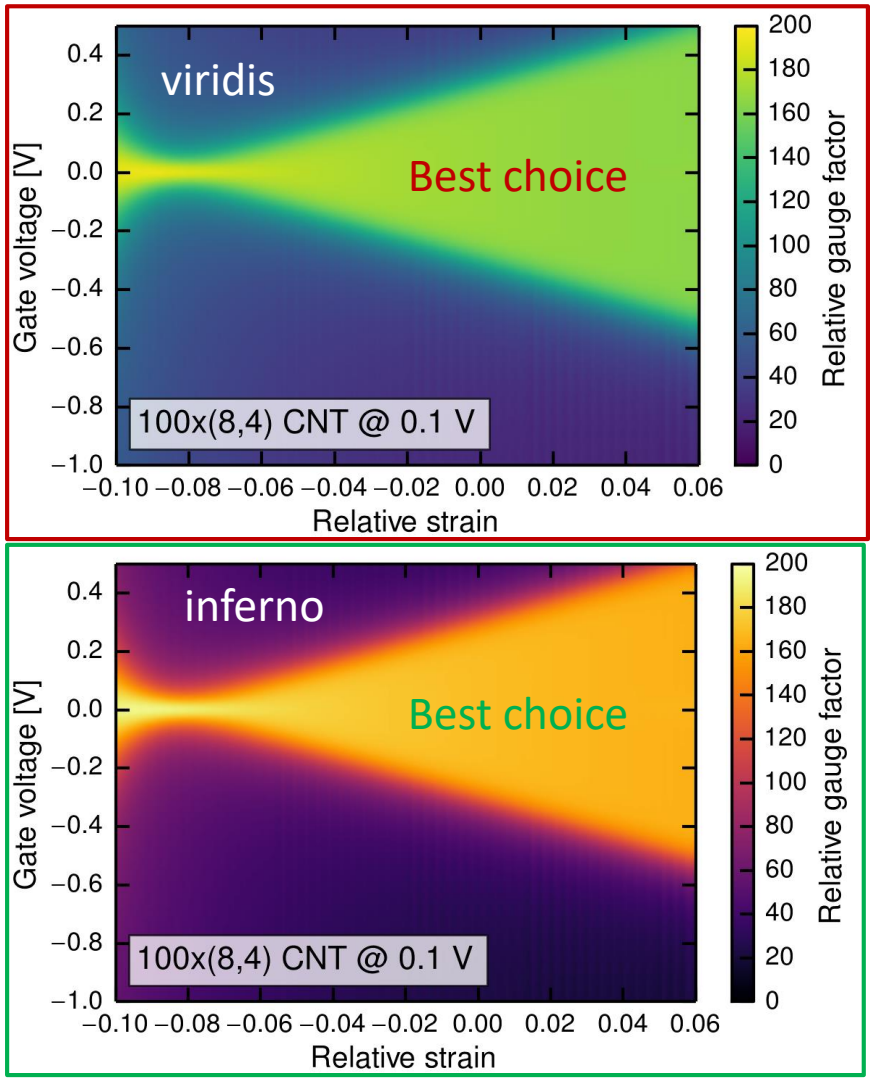


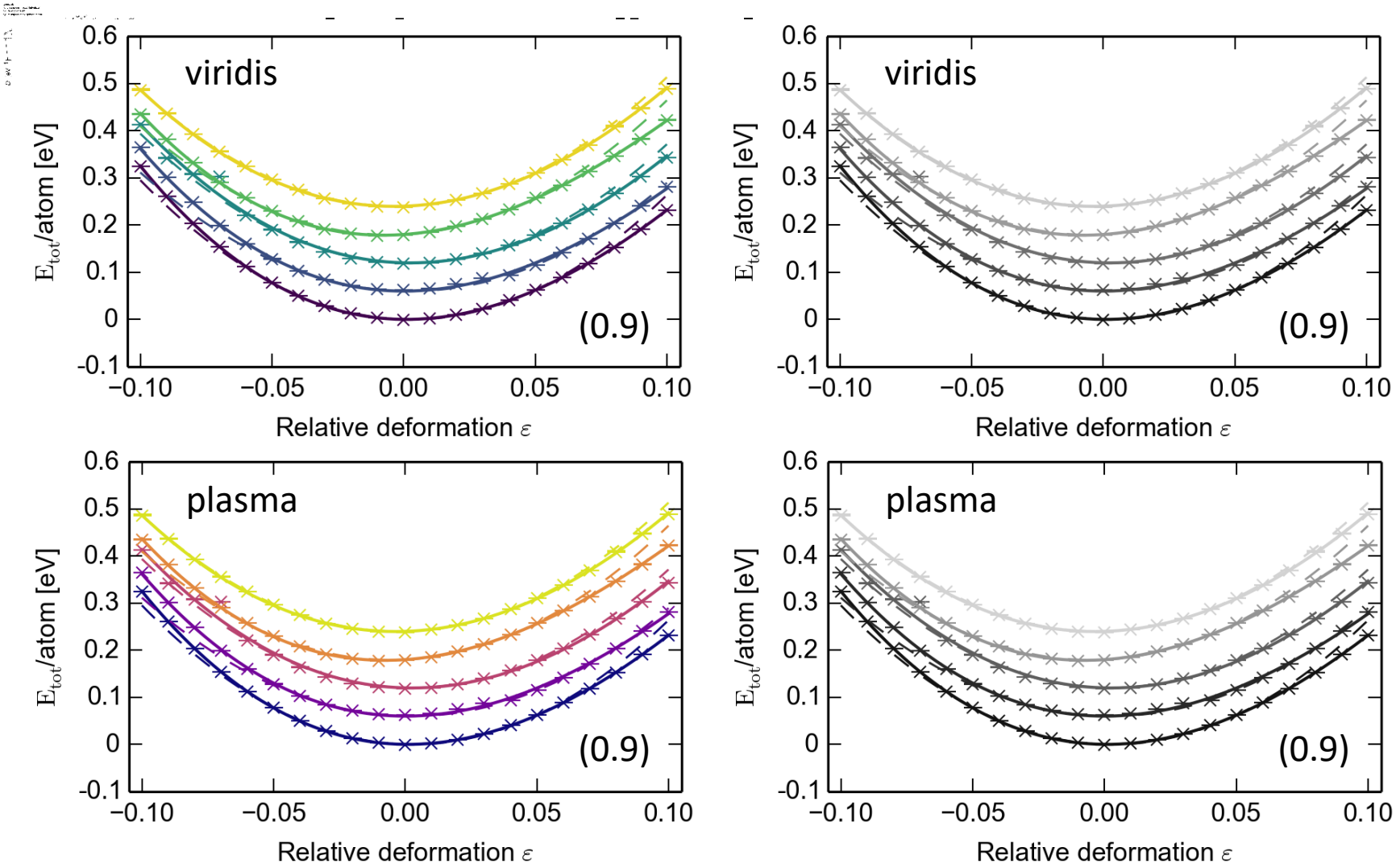
Novel

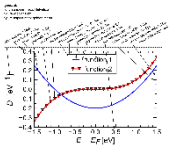




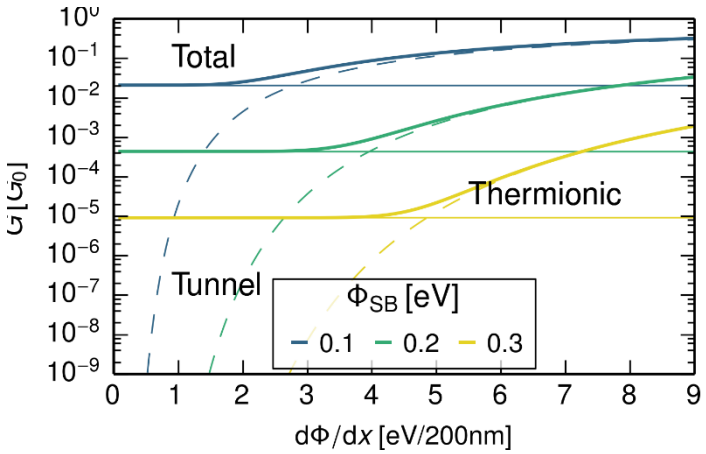

CMRmap*
100x(8,4) CNT @ 0.1 V
Gate voltage [V]
Relative strain
Relative gauge factor
gnuplot**
100x(8,4) CNT @ 0.1 V
Gate voltage [V]
Relative strain
Relative gauge factor

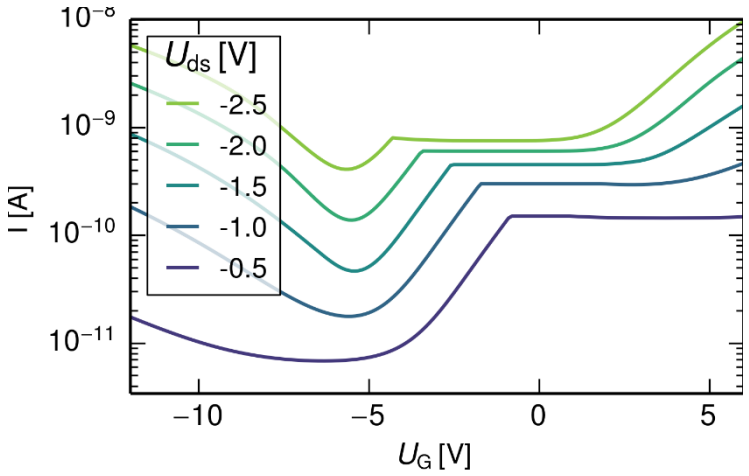

viridis
Best choice
100x(8,4) CNT @ 0.1 V
Gate voltage [V]
Relative strain
Relative gauge factor
inferno
Best choice
100x(8,4) CNT @ 0.1 V
Gate voltage [V]
Relative strain
Relative gauge factor

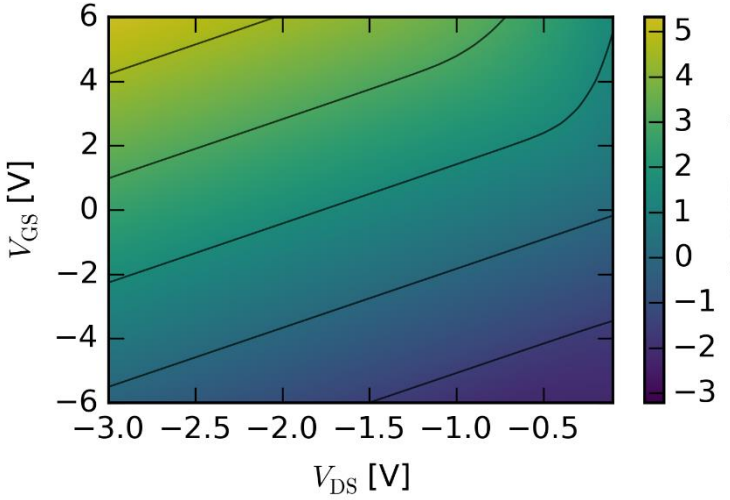

viridis
(0.9)
E_{tot}/atom [eV]
Relative deformation ε
viridis
(0.9)
E_{tot}/atom [eV]
Relative deformation ε
plasma
(0.9)
E_{tot}/atom [eV]
Relative deformation ε
plasma
(0.9)
E_{tot}/atom [eV]
Relative deformation ε

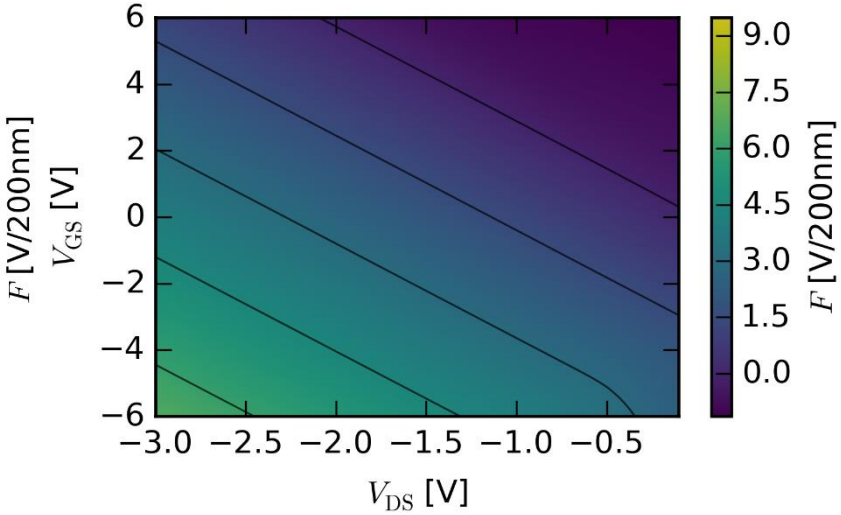


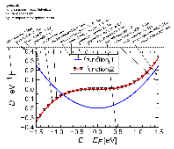
Beispiel aus einer Publikation





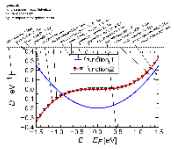






Zusammenfassung

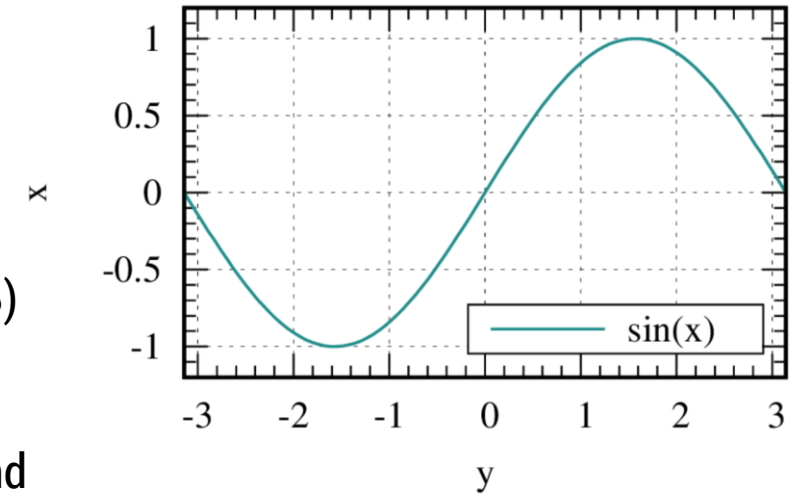
- „Style“-files für optimalen Arbeitsfluss beim Generieren wissenschaftlicher Grafiken
  - Stile: Text, Folien, Poster (einspaltig und zweispaltig)
  - Am besten: Maßvorgabe (Breite) und Größe der Schriftart
- Auswertung und Plotten trennen
- Auf Farbgestaltung achten (Isometrie wichtig?)
  - Inzwischen Nacharbeiten von Matlab:
    „parula“-Palette ist fast isometrisch
  - Nun: Hands-on für gnuplot
    Import der wesentlichsten Farbpaletten möglich –
    www.gnuplotting.org

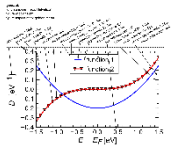


Gnuplot-scripte

- *.cfg: Konfiguration für Grafikexport
- *.pal: Palettendefinitionen
- *.gp: Eigentliches gnuplot-script

1) Erstellen des Verzeichnisses `~/.gnuplot`
2) Extrahieren der Dateien von `gnuplot_templates.zip` (Website) dahin
3) Erstellen eines Arbeitsordners `~/gnuplot_tutorial` o.a.
4) Entpacken des Tutorials dahin (`tutorial.zip`)
5) Befehl `gnuplot < example_plot.gp` ausführen (oder nur `./example_plot.gp`). Datei `example_plot.eps` sollte entstehen und so aussehen*





Nützliche Links

- Gnuplot:
  - http://www.gnuplotting.org/matplotlib-colormaps/
  - http://www.gnuplotting.org/matlab-colorbar-parula-with-gnuplot/
  - https://github.com/Gnuplotting/gnuplot-palettes
    - http://gnuplot.info/demos/

Übungen

1. example_plot.gp so verändern, dass eine .pdf-Datei entsteht (mit Hilfe der Vorlage `one_column_pdf.cfg`)
2. Plotten eigener Datenpunkte (z.B. `data.txt`) zusätzlich dazu
3. 2D-plot in verschiedenen Farbverläufen plotten (Dateien `example2_0.gp` … `example2_2.gp`)
   Tutorial auf website
   http://www.gnuplotting.org/circular-heat-map/