NTNU
Norwegian University of
Science and Technology

Analysis and Simulation of Intermediate Band Solar Cells

Peter Kristoffersen

Master of Science in Physics and Mathematics
Submission date: December 2015
Supervisor: Turid Worren Reenaas, IFY

Norwegian University of Science and Technology
Department of Physics

Sammendrag

Intermediate band solceller (IBSC) er en teknologi som lover høyere effektivitet med forholdsvis enkel struktur sammenlignet med de for tiden beste solcellene på markedet. I dette arbeidet har det blitt gjort videre skritt mot å fastslå vurderingen av materialer for bruk i IBSC. Arbeidet har blitt delt i tre deler: Utvikling og testing av et simuleringsprogram for IBSC; målinger av strølys i luminisensapparatet og luminisensmålinger av IBSC prøver.

Den første delen av avhandlingen tar for seg utviklingen og testingen av et simuleringsprogram for intermediate band solceller som kalkulerer ladningsbærertetthet, generasjon og rekombinasjonsstrømmer, samt mange andre resultater. Programmet bygger videre på doktorgradsavhandlingen til Maryan G. Mayani, men fikser noen feil og legger til støtte for Auger rekombinasjon, noe som aldri har blitt gjort før for IBSC. Bidraget fra Auger rekombinasjon er viktig under høy lyskonsentrasjon, hvor ladningsbærertettheten ofte er høyere.

Fra tidligere arbeid med fotoluminsens gjort ved Institutt for Fysikk ved NTNU ble det oppdaget problemer med det nye utstyret. På grunn av dette har luminisens sjelden blitt brukt som en analysemetode av medlemmene på solcellegruppen på instituttet. Problemet stammer fra uønsket lys i prøvekammeret i fotoluminisensapparatet. Den andre delen av avhandlingen min var å analysere samt å foreslå en metode for å dempe strølyset og effekten det har på resultatet fra målinger. Kilden til strølyset ble bestemt til å være fra lysspredning fra veggene i monokromatoren.

Ettersom det ble klart at den nåværende oppsettet var uegnet til mine planlagte målinger, ble arbeidet forsinket til vi kunne legge en høyfrekvent laser til maskinen, en prosess som ble gjennomført i august 2015.

Den siste delen av oppgaven er å sammenligne luminisensspektra av Cr:ZnS IBSC fra fotoluminisens og elektroluminisens, for å finne ut om en metode er bedre enn den andre på å analysere IBSC. Dessverre har gjentatte forsinkelser gitt meg lite tid til å etablere en eksperimentell prosedyre og å foreta målinger og analyser. Målingene ble gjort ved hjelp av tre forskjellige oppsett,

men resultatene var mangelfulle og ufullstendig og er presentert i appendikset.

Abstract

Intermediate band solar cells (IBSC) is a technology that promises higher efficiency solar cells with comparatively simpler structure than the currently best performing solar cells on the market. In this work, different steps towards the assessment of IBSC materials for use in solar cells have been made. The work has been divided into three parts: Development of a simulation program for intermediate band materials; measurement of stray light in the photoluminescence apparatus; and luminescence measurement on IBSC samples.

The first part of the thesis is a simulation program for IBSC which calculates the carrier concentrations, generation and recombination currents in and between bands. It builds on the doctoral work of Maryam G. Mayani, but fixes a couple of errors and adds calculation of Auger recombination, which in certain materials dominate over Shockley-Read-Hall recombination at higher carrier concentrations. This simulation of Auger recombination has never been done with intermediate band solar cells. The contribution of the Auger recombination is important at higher illumination concentrations where the carrier concentrations are often higher.

From earlier work with photoluminescence done at the Department of Physics at NTNU, problems were noticed with the new equipment. Because of this, luminescence has been rarely used for analysis by the members of the Solar Cell Physics group at the department. The problem stems from stray light in the sample chamber of the photoluminescence device. The second part of my work was to analyze and suggest a way of mitigating the detrimental effects of this light on the results. The source of the stray light was determined to be due scattering on the monochromator walls.

As it became apparent that the current setup was unsuited to my planned measurements, work was delayed until we could add a high-frequency laser to the machine, a process that was completed in the August 2015.

The final part of the thesis is comparing luminescence spectra of Cr:ZnS IBSC obtained with photoluminescence and electroluminescence, to find out if one is better than the other at analyzing intermediate band materials. Unfortunately, repeated delays gave

me little time to establish an experimental procedure and make measurements and do analysis. Measurements were done using three different setups, but the results were lacking and inconclusive and are presented in the appendix.

Preface

This report presents the work done for my master's thesis in the final year of the master's study programme "Applied Physics and Mathematics" (MTFYMA) with focus on applied physics at the Norwegian University of Science and Technology (NTNU). The work has been done at the Department of Physics under the supervision of Turid Worren Reenaas and Mikael Lindgren.

This thesis turned out very different from what I expected. The thesis I started writing had a different focus, different title and different aspirations. I spent the semester preceding my master working on luminescence, and it was my intention to continue doing that. However, equipment defects and delays, as well as other messy stuff that eventually lead to the realization that the main focus of my thesis would have to be relocated to the appendix, where it now lays unrealized.

This meant that what had previously been a side project to work on while waiting for the main course, now had to be expanded to fill a master thesis by its own. Instead of writing about something I'd spent over half a year doing, I would have to write about something completely new. Looking back at what I've written, I feel like I did the best I could with the cards that were dealt me, and I am content with the result, if not with the path that brought me here.

I'd like to acknowledge some of the people who has helped and guided me through this journey. First of all I would like to thank Turid Reenaas for the guidance and help throughout the project; The weekly meetings were invaluable and insightful. And thanks Stanislav Polyakov for teaching me how everything works; whether it's equipment or the university bureaucracy. I'd also like to thank him for making the final year of five at NTNU the most enjoyable of them. Thank you to Mikael Lindgren who stepped in on short notice when I needed help at the end of summer. I'd like to thank Mohammadreza Nematollahi who provided samples for me to test, and offered me assistance. For making food when I was tired without (much) complaining, and for eating the food I made without (much) complaining, I'd like to thank my brother Simeon. And to Thomas Brakkstad, who will probably work on some of the things I've been working on, I wish you more luck than I had.

Abbreviations

CB	Conduction Band
EL	ElectroLuminescence
IB	Intermediate Band
IBSC	Intermediate Band Solar Cell
PL	PhotoLuminescence
QD	Quantum Dot
QFL	Quasi Fermi Level
SC	Solar Cell
SRH	Shockley-Read-Hall
VB	Valence Band

Contents

Chapter 1

Introduction

As the world population increases and the poorest of us seek a higher standard of living, electricity consumption has been steadily rising, increasing by 50% in the last 10 years, with the growth increasing. With an annual electricity production of 24 000 TWh, 80% of this energy is generated from non-renewable resources, most of which contributes to global warming. [1] This is an alarming trend, as not only will continued reliance of these resources damage the climate, but they will eventually run out. It's therefore important to find alternatives to these energy sources that will serve as a sustainable way of providing energy in the future.

A perhaps obvious solution to the problem is solar power. Most current sources for energy stems in one way or another from solar power. Using solar cells to generate electric power directly is cutting out the middlemen we get when generating it from coal, oil or wind. The sun irradiates the earth with 120 PW of sunlight, enough to satiate the world's yearly energy needs in one hour [2].

The generation of electricity from light was first documented by Edmond Becquerel in 1839 in a chemical experiment involving electrodes coated with light sensitive materials. After more than a century of investigating photo-generated electricity, the first modern silicon solar cell was announced by Pearson, Fuller and Chapin in 1954 with an efficiency of 6%. At the time, the silicon production industry was immature, and the cells were mainly used for space probes [3]. Since then, the technology behind solar cells have developed steadily, enabling higher efficiency devices at lower prices.

Interest for solar power has also moved from space technologies to everyday life as the sustainability of conventional energy sources have been cast into doubt. Today the price of solar power has only reached grid parity in several countries [4].

The invention of the silicon semiconductor solar cell started what we now generally refer to as the first generation of solar cells. The community often refers to three generations of solar cells, divided as shown in figure 1.1. First generation solar cells are crystalline cells that have reached record efficiencies of 25% [5]. The second generation of solar cells refer to solar cells made with techniques and materials aimed at significantly decreasing the cost of production at the expense of efficiency. Table top calculators and other low power consumer products often use these cells. The second generation solar cells have a record efficiency of 20% [5]. The third generation solar cells aims at exceeding the efficiency of the previous generations, with current record efficiencies of 46% [5].

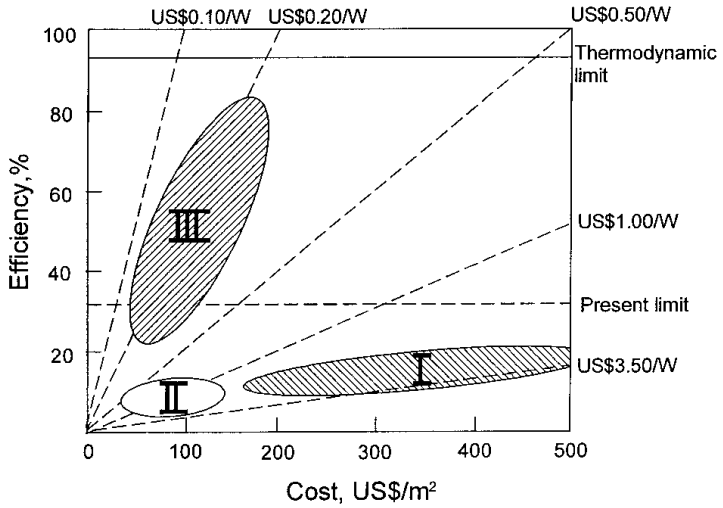

Figure 1.1: The cost and efficiency of the first-, second- and third-generation solar cells. Taken from ref. [6].

One of the approaches to a third generation solar cell is the intermediate band solar cell (IBSC), which is one of the main topics of this thesis. Specifically, a simulation program for analyzing properties of IBSCs has been made and is presented. It aims to investigate the effect of non-radiative recombination at higher light concentrations. Experimental work on photoluminescence of IBSCs

produced at NTNU was conducted, but the data was deemed too inconsistent for analysis and was not included in the main text of the thesis, and is instead included in appendix A. In preparation for the luminescence analysis, stray light previously detected in the luminescence apparatus was characterized to determine if the current setup was suitable for further measurements.

The theory behind solar cells, how they work and the challenges limiting efficiency is given in chapter 2. The chapter will also explain more about third generation solar cells, with focus on IBSCs. At the end it describes a theoretical method of determining the efficiency of solar cells, the "detailed balance" model, which is used in the simulation program.

In chapter 3, Methods, I will go through the implementation of the simulation program in detail, as well as explain about the luminescence equipment, and how the stray light characterization was done. The results of the thesis, with outputs from the simulation program and stray light measurement is presented in chapter 4.

Chapter 2

Theory

This chapter will explain how solar cells turn sunlight into electricity, while detailing the challenges we face when trying to optimize the efficiency. It will then explain about the different generations of solar cells, what defines them, and specific technologies utilized in each of the three generations. Lastly, I will introduce the reader to the concept of "detailed balance", a model for calculating the theoretical efficiency of a solar cell. Detailed balance is used in the simulation program in chapter 3.1.

2.1 The basics of a solar cell

A solar cell, also called a photovoltaic cell, is a device that absorbs sunlight and transforms some of the energy from the absorbed photons directly into electric energy. The different parts of a simple solar cell is shown in figure 2.1. The main part of a solar cell consists of two regions of semiconductor material doped so that one has a surplus of positive carriers, holes, and the other has a surplus of negative carriers, electrons. These two regions are called the p-type and n-type layer, respectively. The boundary between these two layers is the pn-junction. On the front and back of the solar cells are contacts. The back contact is usually a solid piece of metal, but the front contact, to let in light, is often a grid of metal "fingers" to collect the carriers.

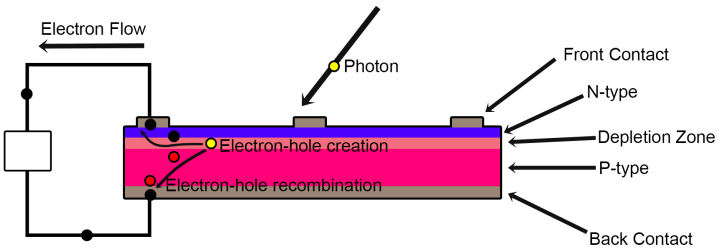

Figure 2.1: Schematic of the basic parts of a solar cell. An incoming photon is absorbed in the depletion region with the creation of an electron-hole pair. The electrons are pushed by the internal potential field to the front contact. After doing useful work in an electrical circuit, the electron recombines with a hole in p-type material.

The photovoltaic process is shown in figure 2.2. When a photon is absorbed by the solar cell, it has the chance of energizing a carrier. If the energy of the photon is high enough, this carrier can jump into the conduction band of the semiconductor. The difference in energy between carriers in their "rest" state and the conduction band is called the band gap, which for silicon solar cells is around 1.1 eV. The excited carriers will now cross the pv-junction, to a lower energy state. This happens due the the electric field that builds up at a pn-junction. Holes will collect in the p-region and electrons will collect in the n-region. This separation of charge gives rice of a voltage that can drive current from one contact to the other. Electrons can now exit the solar cell through the n-side contact, do electrical work,, and finding itself on the p-side side of the solar cell again. Photons with lower energy than the band gap are unable to kick a carrier across the gap, and will not contribute to the electric energy produced [7, 8].

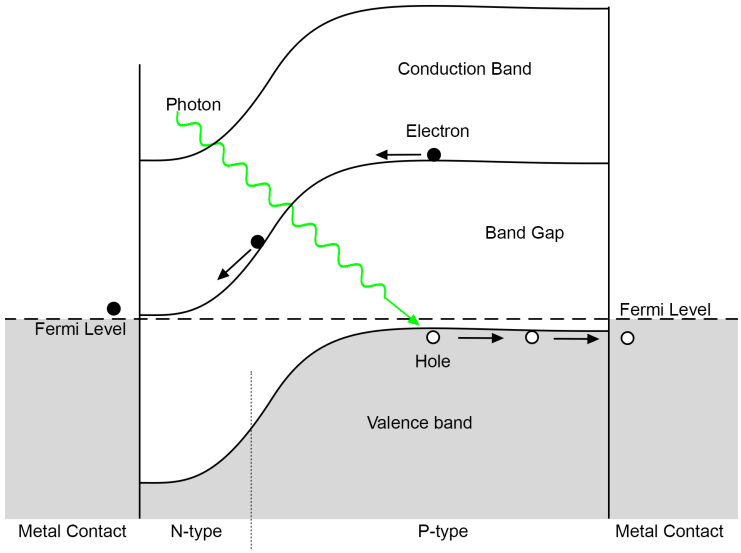

Figure 2.2: Schematic of the band gap structure of a solar cell. An incoming photon will excite an electron from the valence band which is full of electrons into the conduction band which is normally empty. The excited electron is moved towards the left due to the potential change in the pn-junction. The electron will travel through the metal contact and a circuit before filling a hole on the right side of the figure. In this figure, the solar cell is short-circuited, so the metal contacts are at the same energy level.

2.2 Loss mechanisms in solar cells

Solar cells are not 100% efficient. Numerous loss mechanisms exist that decrease the energy available for extraction, as illustrated in Figure 2.3.

If an incoming photon has energy less than the band gap of the semiconductor there are no allowed states for electrons to be excited to, thus the photon is not absorbed. The semiconductor is effectively transparent for photon energies less than the band gap. This means that the part of the spectrum with lower energy than the band gap can not be used for energy production. Since ideally every photon would be converted to an electron, this leads to a reduced current. This loss can be reduced by lowering the band gap of the solar cells, but as the electrical power extracted from a solar cell is $P = IV$,

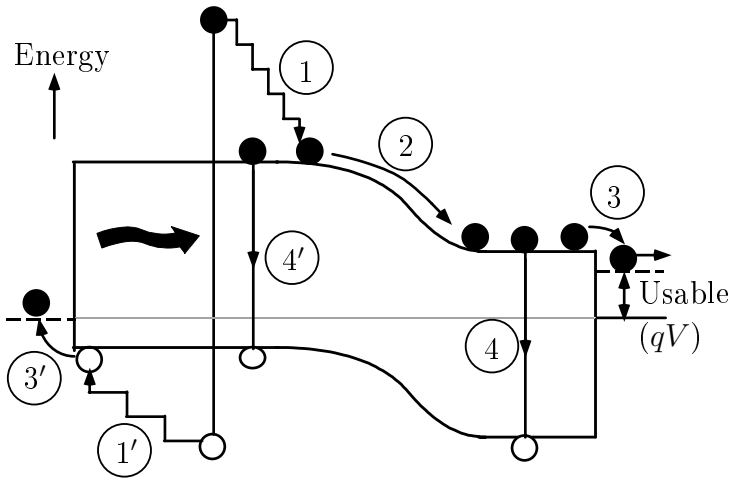

Figure 2.3: Loss mechanisms in a solar cell. (1) Thermalization loss. (2) Junction loss. (3) Contact loss. (4) Recombination loss. Taken from ref. [9].

both current and voltage must be optimized.

Conversely, if the photon has energy higher than the band-gap, the excess kinetic energy is lost in collisions with the crystal lattice. This loss can be lowered by increasing the band gap, increasing the cell voltage, but decreasing the current. Single junction solar cells are therefore tuned to one specific wavelength of light where these losses are minimized [9].

Recombination is the process in which an excited electron falls down to the valence band again. This is detrimental when it happens before the carrier has been through the electrical circuit. This loss mechanism can be divided into radiative recombination and non-radiative recombination, as shown in figure 2.4. Radiative recombination, or band-to-band recombination is the process where an electron releases energy as a photon. This process is unavoidable, but does not have to be detrimental, as photons emitted this way can be reabsorbed. Non-radiative recombination is a process where an electron releases energy as heat in collisions with the crystal lattice. Different types of non-radiative recombination exist.

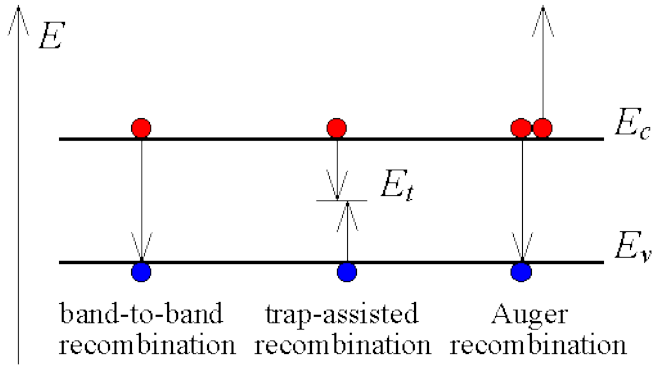

Figure 2.4: A figure of the three types of recombination. Band-to-band recombination is radiative, and generates a photon with the band gap energy. Trap-assisted recombination, often referred to as Shockley-Read-Hall (SRH) recombination, are recombination processes via states in the band gap. Auger recombinations are recombination processes where band-to-band recombination processes transfers energy to another carrier which then gradually releases this energy in lattice collisions. Figure taken from ref. [10].

One type of non-radiative recombination is Shockley-Read-Hall (SRH) recombination. This type of recombination happens when there are defects in the crystal structure, creating available states within the band gap. The recombination happens when first one carrier is trapped in the defect state, and then the other type is captured a time later. The rate of recombination is highly dependent on the position of the defect state in the band gap, with states near the middle of the gap most detrimental. SRH recombination is avoidable by creating high quality crystals with few defects [8, 11].

Auger recombination is a process that involves three carriers, as opposed to the two other types of recombination we have seen. There are many Auger processes, but the simplest one is shown in figure 2.4. An electron in the conduction band falls to the valence band and recombines with a hole. The energy released is given to another electron in the conduction band which will fall back to the conduction band edge through thermalization losses. Because it involves three carriers, the recombination rate tends to be proportional to the cube of the carrier concentrations, meaning

Auger recombination can dominate the other processes at high illumination. Auger processes can also generate excited carriers, in which the opposite of an Auger recombination happens: An energetic electron collides with an electron in the valence band, causing both to end at the conduction band edge[11].

There are also losses associated with the junction and the contacts. From figure 2.3, you can see that process 2 and 3 are voltage drop in the pn-junction and contacts respectively. With right choice of materials, these losses can be reduced, but not eliminated. When producing current, one also has to take into account voltage losses due to electrical resistance in the cell[11].

2.3 Third Generation Solar Cells

Third generation solar cells is a collective term for solar cell technologies that both enhance efficiency and decrease cost compared to conventional SCs, illustrated in Figure 1.1.

Many proposed ideas for enhanced efficiency solar cells exist, but most of them can be divided into three categories.

Multiple electron-hole pair solar cells work by creating more than one electron-hole pair from high energy photons. An example of this type of cell is the impact ionization cell, where the kinetic energy of a hot electron collides with the lattice electrons and creates an additional electron-hole pair.

Hot carrier solar cells aim to extract the carriers before they can relax to the band edge. In normal materials, this relaxation is very fast, in the order of picoseconds. Research on hot carrier cells is therefore focused at either slowing the cooling of the carriers, or speeding up the extraction [9, 6].

Finally we have the multiple junction solar cells. In its simplest form, these can be thought of as a series of solar cells with gradually decreasing band gaps, as illustrated in Figure 2.5. Lower energy light will pass through the first layers until it reaches a cell with low enough band gap. There are a couple of ways for creating such cells. Cells can be layered in stacks or spectrally sensitive mirrors can split the light with different wavelengths onto separate cells[9].

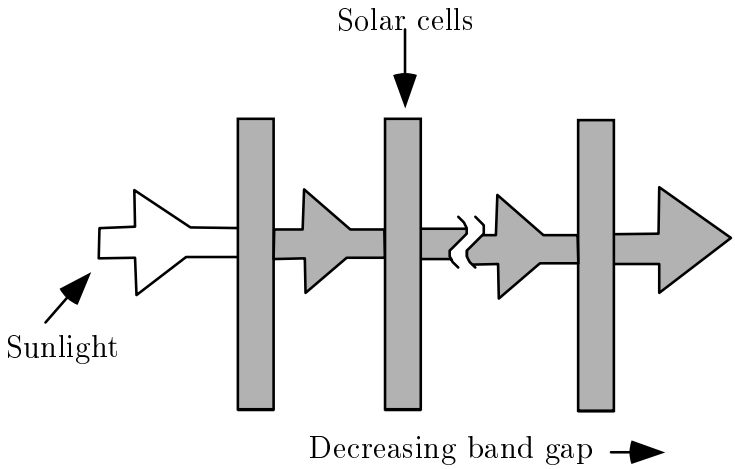

Figure 2.5: Schematic drawing of tandem cell approach. Taken from ref. [9].

The intermediate band solar cell (IBSC) design is a type of multiple band gap solar cell. Like the other third generation solar cells, IBSCs more efficiently extracting the energy of photons that wouldn't match the band gap of a conventional cell. It does this by introducing at least one additional energy band, as illustrated in Figure 2.6. Light with energy $h\nu \geq E_g$ can still carry electrons from the VB to the CB, but photons with lower energy can use their energy to partially lift an electron to or from the intermediate band (IB) [12].

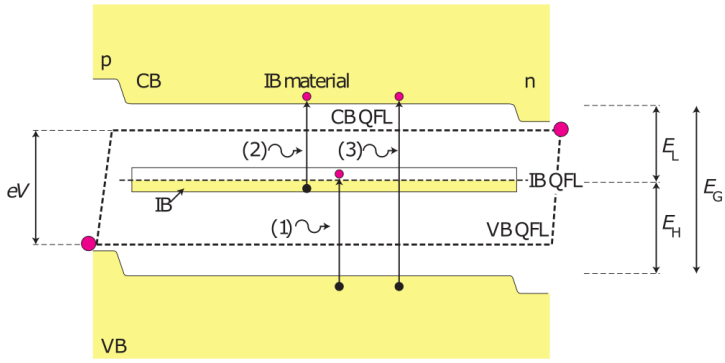

Figure 2.6: Band diagram for an IBSC showing band gap E_G and sub-band gap E_L and E_H. Also included are the CB, VB and IB quasi-fermi levels (QFL). Taken from ref. [12].

One can think of this as a combination of three normal solar cells with different band gaps with the restriction that the current through the two smallest band gaps must be equal, to maintain a constant number of electrons in the IB. Detailed balance calculations put the limiting efficiency of a single IB cell at 63% under concentrated light, compared to 41% for a single-gap solar cell under the same conditions [12].

Research has been ongoing for more than 10 years, but scientists still have not been able to create a commercial-grade IBSC. Researchers have proposed many ways of creating the needed intermediate band, and analysis of different materials is needed to investigate their properties [12]. I will give a short introduction to three methods for making IBSCs.

Quantum dot IBSC use layers of quantum dots to form the intermediate band. A small quantum dot (QD) forms a quasi 0D system where the carriers are confined in all directions. If many of these QDs are placed close to one another, the wave functions of neighboring dots can overlap, the energy levels will couple and we get a miniband [13]. This is the most studied IBSCs, but there are several challenges. The growth method used relies on lattice mismatch, and a very high concentration of quantum dots are needed to diminish SRH recombination [14].

In highly mismatched alloy (HMA) IB materials an insertion of a small amount of an elemental impurity into a semiconductor host material causes a splitting of the conduction band, where the lowest of the splits take on the role as the intermediate band [15].

A third type of IBSC is the deep level intermediate band solar cell (DL-IBSC), where the IB is formed by doping suitable semiconductors with a high consentration of a dopant, resulting in a deep level in the bandgap [14].

In addition to the loss mechanisms described in the previous section, IBSCs are subject to further factors that limit efficiency. One example is overlapping absorption coefficient. Since each sub-band gap absorbs photons at different wavelengths, you don't want one of the larger band gaps to "steal" photons from the smaller one. If the absorption spectrum for each band gap overlaps, there will be some loss compared to the optimal case with no overlapping

absorption[16].

2.4 Detailed balance

One of the models for calculating theoretical solar cell efficiency is the detailed balance model. It was used by Shockley and Queisser in 1961 to calculate the limiting efficiency for single band solar cells [7]. This section on detailed balance is mostly based on references [8, 7, 11].

The theory behind detailed balance requires every process in the system must be in an equilibrium with the inverse process, as long as the system itself is in thermodynamic equilibrium. Solar cells don't only absorb photons, but also emit them. For instance, in the dark, with no applied voltage, the absorbed and emitted photon flux is the same. Since every absorbed photon cause an electron hole pair, the current from a solar cell without any non-radiative radiation is

$$J_{\text{net}} = J_{\text{abs}} - J_{\text{rad}} = q(G - R)$$

Where G and R are generated and radiated photons respectively.

The photon flux of photons with energy E hitting a point on the surface from a black body at temperature T_s spanning an angle F ($F = \pi$ means the flux is received from a hemisphere) is

$$b_s(E; T_s) = \alpha(E)\frac{2F}{h^3c^2}\left(\frac{E^2}{\exp(E/k_BT_s) - 1}\right),$$

where $\alpha(E)$ is the absorptivity of the cell at photon energy E.

The flux of photons of energy E emitted from a cell with chemical potential $\Delta\mu$ and temperature T_a is given by [7],

$$b_e(E, \Delta\mu; T_a) = \alpha(E)\frac{2n_s^2}{h^3c^2}\left(\frac{E^2}{\exp([E - \Delta\mu]/k_BT_a) - 1}\right).$$

If the mobility of the carriers is sufficiently large, the chemical potential will be equal to the voltage across the cell, with $\Delta\mu = qV$ if you are using SI units for the chemical potential.

In an IBSC, we have optical generation and recombination from three different band gaps. If we do a couple of assumptions and assume the absorption is 1 for only the non overlapping parts of the spectrum and 0 for the rest, and if we disregard photons from ambient sources and assume all generated current is due to solar photons we get

$$\begin{aligned}
G_{vc} &= \int_{E_g}^{\infty} b_s(E, T_s)\,\mathrm{d}E \\
G_{vi} &= \int_{0}^{E_{ib}} b_s(E, T_s)\,\mathrm{d}E \\
G_{ic} &= \int_{E_v i}^{\infty} b_s(E, T_s)\,\mathrm{d}E \\
R_{cv} &= \int_{E_g}^{\infty} F_e b_e(E, \Delta\mu, T_a)\,\mathrm{d}E \\
R_{iv} &= \int_{E_v i}^{E_g} F_e b_s(E, \Delta\mu, T_a)\,\mathrm{d}E \\
R_{ci} &= \int_{E_i c}^{E_v i} F_e b_s(E, \Delta\mu, T_s)\,\mathrm{d}E
\end{aligned}$$

where T_s and T_a are the temperatures of the sun and the cell respectively. The subscripts v, i and c correspond to the valence, intermediate and conduction band, with the order specifying the direction of electron flow. The energies E_{vi} and E_{ic} are the energy difference between the valence and intermediate; and the intermediate and conduction band, respectively. In this example, $E_{vi} > E_{ic}$. The factor F_e in the radiative terms, analogue to F in the generation terms, is of the form

$$F_{e,\mathrm{rad}} = 2\pi \int_0^{\theta_c} \left(1 - \exp\left[\frac{-2\omega\alpha}{\cos x}\right] \cos x \sin x\right)\,\mathrm{d}x$$

to account for reabsorption of emitted photons, so-called photon recycling [17].

If we know all the relevant material properties, we can then calculate the effect of this solar cell by multiplying the electron current from the valence band to the conduction band with the elemental charge

$$P = q(G_{vc} - R_{cv} + G_{vi} - R_{iv}).$$

The contributions for the ic band are already counted in the vi terms, since there is always a net zero current into the intermediate band.

To support more complex models, we only need to modify and add terms in this last equation. For example adding non-radiative terms,

$$P = q(G_{vc} - R_{cv,\mathrm{rad}} + G_{vi} - R_{iv,\mathrm{rad}} - R_{cv,\mathrm{SRH}} - R_{iv,\mathrm{SRH}}).$$

This is how SRH and auger recombination is added in the simulation program in the chapter 3.1.

Chapter 3

Methods

In this chapter I will explain about the methods and equipment used in the project. Chapter 3.1 will familiarize the reader with how the simulation program works. Chapter 3.2 will detail the luminescence equipment and how it was used.

3.1 Simulation of IBSC with recombination

I have written a computer program to simulate various elements of an IBSC under illumination. The complete program, along with sample inputs can be found in appendix B.1 and B.2 respectively.

The program itself is based on a previous work by Maryam Gholami Mayani. The new program is much faster thanks to a different implementation. Additional features have been added, such as the possibility to simulate auger recombination. Additionally, errors have been corrected, and the new program should be more readable and reliable.

3.1.1 Assumptions and simplifications

To simplify the simulations, some assumptions and simplifications have been made. Below follows a non-exhaustive list of them.

Complete ionization:
The simulation assumes that all of the dopant atoms are ionized. This means that the simulation will be less accurate at lower temperature conditions.

Non overlapping absorption:
The best case scenario is assumed, where the incident light generates carriers in the largest possible band-gap. In reality, light with energies $h\nu > E_g$ can still be absorbed by the smaller sub-band gaps.

Nondegeneracy of states:
The simulation makes the assumption that all states are non-degenerate. If the simulated material has degenerate states, the recombination equations in particular should be reassessed for correctness.

All states in a band have the same energy:
In reality, as an energy band fills, new carriers will have to be generated into higher energy states. In the simulation, bands are assumed to be composed of a number of states, all with the band edge energy. The error introduced by this assumption is larger for when a large portion of the states are filled.

No generated current from ambient photons:
In reality, not only photons from the sun will generate a current in a solar cell, but also the photons from the environment, which is often assumed to be a black body at room temperature. This contribution is assumed small in comparison to the sun's contribution.

3.1.2 Implementation details

The simulation part of the program consists of three parts, where the second one is the most computationally intensive. The parts are marked with comments in the code. Variables and objects from the code are typeset in monospaced font, like so: `variable_name`.

Part 1

Part one of the code calculates the carrier concentrations and Fermi energies of the bands in thermal equilibrium. The program achieves this by solving Poisson's equation for the material.

$$\nabla^2\varphi = \frac{\rho_f}{\varepsilon}$$

In thermal equilibrium, the left hand side of the equation is 0, since this is equal to the divergence of the electric field, $\nabla\cdot\mathbf{E}$. The correct equilibrium Fermi energy will be the one that gives zero free charge, ρ_f.

The correct value for the equilibrium Fermi energy is found by binary search, a technique often used for finding items in a sorted list. The program starts out by setting a lower and an upper bound for the possible answer. It then calculates the right hand side of the Poisson's equation using the value in the middle of the bounds, disregarding the ε. If the answer is positive, we have to many positive carriers, and the value we tested is too large. The upper bound is then changed to the value we just used, since we now know it, and all higher values, are too large. If the result from the Poisson's equation is negative, we update the lower bound. This assumes that the function $0 = \rho_f(E_0)$ is monotonic and has only one root. From the definition of the function, we can see that this is always the case, as all the terms of the free charge ρ_f increase with increasing E_0.

Every iteration of the binary search increases the precision of the answer by a factor of 2. The number of iterations is specified in the variable `N_iter`. The binary search will find the correct value to a with a precision of $\pm E_g 2^{-(N_{\text{iter}}+1)}$.

Part 2

Part two of the code involves solving the continuity equation for the intermediate band. The charge entering the intermediate band must be equal to the charge exiting it. We do this for each of a number of `V_num` voltages over the cell.

The code uses the same binary search algorithm as in part one, but modified so it can search for all `V_num` voltages at the same

time. Instead of finding the correct value of the Fermi energy, since there is now a Fermi level split and a single Fermi energy does not exist, the value that is searched for is `Vshift`, which is the how much the Fermi level split is offset from the valence band edge, see figure 3.1.

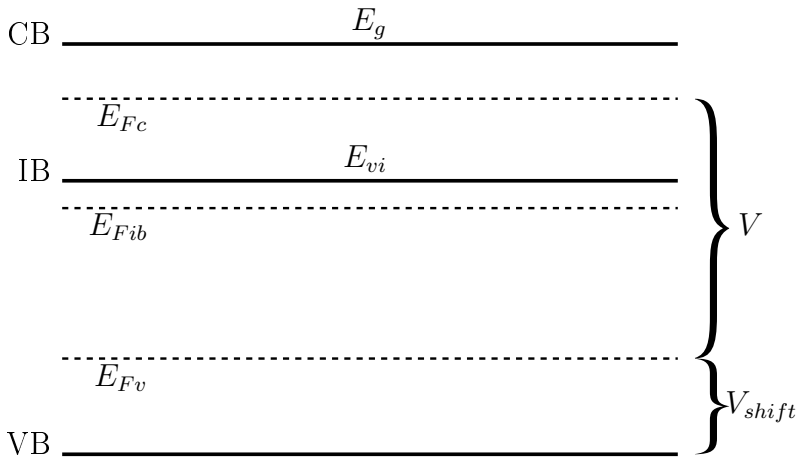

Figure 3.1: The figure shows some of the values the program use in its calculations. The value V_{shift} is varied until the electric flux into the intermediate band is zero.

The continuity equation is evaluated for each voltage every iteration of the binary search. The continuity equation for the intermediate band is defined as

$$0 = G_{vi} + R_{ci} - G_{ic} - R_{iv}$$

Where G represents photo-generated current, and R represents recombination current, and the subscripts represent which bands are involved, i.e. a subscript ic is current from the intermediate band to the conduction band.

Although different mechanisms for exciting carriers exist, only photo-generated carriers are calculated in the program, and is assumed generated by photons from a black body. This is calculated as

$$G = \frac{x}{46050}\frac{2n_r^2}{h^3c^2}\int_L^H \frac{E}{\exp\left(\frac{E}{kT}\right)-1}\,\mathrm{d}E \int_0^{\theta_c}\left(1-\exp\left[-\frac{2w\alpha}{\cos\theta}\right]\right)\cos\theta\sin\theta\,\mathrm{d}\theta$$

Where x is the concentration factor and n_r is the refractive index of the material. h and c are the Planck's constant and the speed of light respectively. The first integral and the prefactor gives us the number of generated carriers from photons with energies between L and H. kT is the Boltzmann constant multiplied by the black body temperature. The second integral gives us the fraction of photons actually absorbed. θ_c is the critical angle for internal reflection, w is the width of the cell and α is the absorption coefficient.

While there is only one term for generated current, there are three terms for the recombination current.

First is radiative recombination, `R_vi` and `R_ic` in the code. It is calculated in much the same way as the photo-generated current, but without the concentration factor $x/46050$, and with a correction to the first integral

$$\int_L^H \frac{E}{\exp\left(\frac{E-\mu}{kT}\right) - 1} \, \mathrm{d}E$$

Where the added μ is the chemical potential of the band gap, defined as the Fermi level split between the conduction and valence band.

The second term of the recombination is the Shockley-Read-Hall recombination due to trap states in the band gap. The recombination rate from a state a to a state b is calculated in the code as [8]

$$SRH_{ab} = \frac{n_a p_b - n_{a0} p_{b0}}{\tau_p(n_a + n_t) + \tau_n(p_b + p_t)} \tag{3.1}$$

Where n_a and p_b are electrons in state a and holes in state b respectively. The 0 subscript indicates the thermal equilibrium values. τ_p and τ_n are lifetime parameters for the traps. n_t and p_t are parameters that depend on the trap energy level, and are defined as[18]

$$\begin{aligned} n_t &= N_a \exp\left(\frac{E_t - E_a}{kT}\right) \\ p_t &= N_b \exp\left(\frac{E_b - E_t}{kT}\right) \end{aligned} \tag{3.2}$$

Where N_a and N_b are the carrier effective densities of state for state a and b respectively, and E_a and E_b are the energies of each state. E_t is the energy level of the state. In the sample input file A and B, see appendix B.2, this energy level is set to be approximately in the middle of each of the two sub-band gaps, where SRH traps are most detrimental.

The lifetime parameters τ_p and τ_n are calculated as

$$\tau_p = \frac{1}{v_n \sigma_n N_t}$$
$$\tau_n = \frac{1}{v_p \sigma_p N_t}$$

Where v_n and v_p are the mean thermal velocities of the electrons and the holes respectively, σ_n and σ_p are the capture cross sections, and N_t is the density of trap states.

The calculation of SRH recombination can be turned off by setting `calculate_SRH = False` in the parameter file.

The last term in calculating the recombination rate is recombination due to the Auger effect. For the intermediate band system, six simple recombination paths exist due to the Auger effect. The different recombinations are illustrated in figure 3.2.

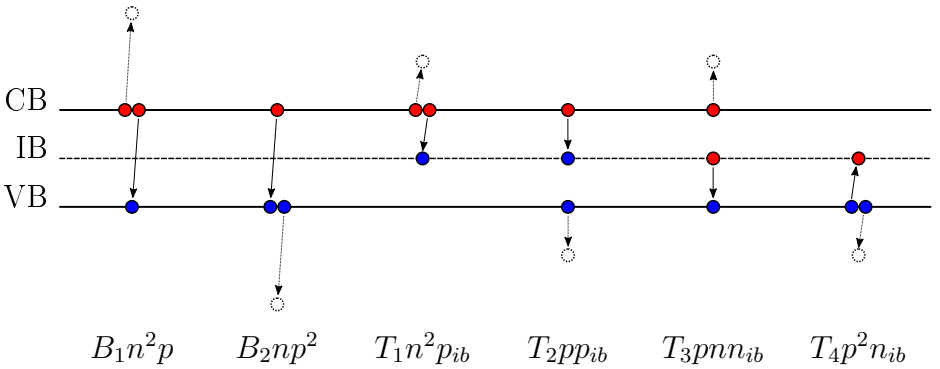

Figure 3.2: This figure shows the six auger recombination processes simulated by the program in a band diagram. Recombination is indicated by an arrow from an electron (red) to a hole (blue). Arrows to unfilled states indicate how the third carrier in the process will move.

The calculation of auger recombination can be turned off by setting `calculate_auger = False` in the parameter file.

Since Auger processes involve at least three carriers, the recombination rates have the form

$$R = Tn^2p \tag{3.3}$$

or

$$R = Tnp^2$$

depending on whether it's a two-electron or a two-hole process. T is a material specific constant.

Although it is possible to set the six recombination constants separately, a shortcut exists to set all of them to the same value, as shown in appendix B.2.

In the previous version of the program, there was no support for simulating Auger recombination, but because the local carrier density in certain types of IBSC is very high, e.g. in quantum dot IBSC, one may expect Auger processes ($\propto n^3$) to dominate over SRH ($\propto n^2$) and radiative recombination ($\propto n$)[19]. There are concerns that Auger recombination might be a limiter to the efficiency of concentrator solar cells for this reason [20].

Part 3

The last part of the program uses the calculated values for each cell voltage to calculate all the additional data that can be useful, such as max power point and efficiency. It also calculates generation and recombination rates that didn't have to be calculated in part 2.

The output of the program is a series of values for many cell characteristics for each cell voltage.

The output variable called `errors` is a bitmap of errors the code encountered for this cell voltage. Only values that have corresponding error 0 can be trusted to be correct. These errors correspond to physical impossibilities. For example, if a negative electron concentration is needed for charge conservation to hold. The errors that can be encountered are specified in table 3.1. To figure out what error was encountered, simply convert the error value to binary and compare with the aforementioned table. For

Table 3.1: Description of errors that can be returned from a calculation.

Name & value	Description
`err_negative_current` $2^0 = 1$	The current through the solar cell would have to be negative for the continuity equation in the IB to hold.
`err_n_ib_negative` $2^1 = 2$	For the continuity equation in the IB to hold, the IB would need to have a negative amount of filled states.
`err_n_ib_too_large` $2^2 = 2$	For the continuity equation in the IB to hold, the IB need more than 100% filled states.
`err_mu_vi_too_large` $2^3 = 4$	The chemical potential of the *vi* band gap is larger than the difference of energy of the states.
`err_mu_ic_too_large` $2^4 = 16$	The chemical potential of the *ic* band gap is larger than the difference of energy of the states.

example, an error of $18 = 10010_2 = 2^1 + 2^4$ will correspond to the errors `err_n_ib_negative` and `err_mu_ic_too_large`.

Some values will not be calculated when an error is encountered. Other values will be calculated, but will not be correct.

The output of the program is automatically saved in a file matching the name of the input file, in the format(s) specified. To specify an output format, use the command line option `-f`. For example, to save the output in text and excel formats, use `ibsimulator.py inputfile.inp -f txt xls`.

3.1.3 Input files

To analyse the results from the band filling simulator, I have decided to focus on a couple of different input files, some of which are included in appendix B.2. All of the files are variations on the same base solar cell with a base material similar to silicon. The

base file, A.inp, calculates both SRH and Auger recombination. File A0.inp has very few states in the IB, simulating what is close to a conventional solar cell. File Ared.inp has reduced Auger recombination for the Auger processes involving the IB. File B.inp and C.inp do not calculate Auger and SRH recombination respectively, and are used in comparison with the old code.

The values used for the physical constants are by necessity partly educated guesses. Many of the values have never been experimentally measured, and where unavailable, similar constants from other materials are used. The doping level has been chosen so that the intermediate band is half filled under no illumination. This is achieved by setting $N_d = N_{ib}/2$, as the electrons from the donors fill half the IB states [17]. The number of intermediate band states, $N_{ib} = 5 \times 10^{17}\,\mathrm{cm}^{-3}$ is a reasonable concentration for current quantum dot based IBSC; and even higher concentrations are possible with bulk IB materials [21]. More information about how the parameters were chosen can be found in the sample inputs in appendix B.2.1.

The IB energy level is chosen arbitrarily at $E_{ic} = 0.21\,\mathrm{eV}$, making $E_{vi} = 0.91\,\mathrm{eV}$. This relatively large difference should prevent a big overlap in the absorbed photon frequencies. With the non overlapping photon absorption assumption, if the sub-band gaps were too similar, the smallest gap would only be able to absorb the small fraction of light with photon energies that fell in between that of the two gaps.

3.1.4 Changes compared to previous work

As mentioned before, the code is based on work by M. G. Mayani. In this section I will lay out the main changes I have made.

The most immediate change is the change of programming language. The original code was written in Matlab, while the new code is written in Python. While by themselves Python and Matlab are comparable when it comes to speed, I chose to port the code due to mainly two reasons. The first is that Python is open source and free. The second is that Python offers more advantages when it comes to object orientation. It's comparatively hard to create small functions in Matlab to do repetitive tasks.

Another obvious change is the added support for calculating Auger Recombination. The expressions used to calculate the recombination from auger processes is taken from "Recombination in Semiconductors"[11], and has the very simple form seen in (3.3).

The code has also got a solid speedup due to a general cleanup, as well as more efficient searching in the new code. As an example, when running case A and C through the old code both more than 24 hours, while the new script takes less than 20 minutes. This speedup is something I wanted from the start to make the script more usable. Another thing that will aid in usability is that the script automatically detects which sub-band the SRH traps are in; while the old program had two versions, one for each circumstance. The new script will also automatically deal with cases where $E_{ic} > E_{vi}$, which would have been hard-coded in the old version.

The code for calculating SRH recombination has been changed. While looking over the old code, I noticed some errors that I decided to fix. SRH recombination is often calculated with the following expression[8]:

$$\begin{aligned} \tau_n &= (v_n \sigma_n N_t)^{-1} \\ \tau_p &= (v_p \sigma_p N_t)^{-1} \\ n_t &= n_i \exp\left(\frac{E_t - E_i}{kT}\right) \\ p_t &= n_i \exp\left(\frac{E_i - E_t}{kT}\right) \\ U_{SRH} &= \frac{np - n_i^2}{\tau_n(p + p_t) + \tau_p(n + n_t)} \end{aligned}$$

There are two errors in the old code. The first is that the signs of the numerator in the exponent of n_t and p_t are switched around in the calculation of SRH recombination between the intermediate and valence band. It's correct for the calculation between the valence and conduction band.

The second error is that the old code uses thermal equilibrium values as if they are intrinsic values. In the first part of the program, equilibrium values are calculated. The old code uses an i subscript

instead of my 0, and seems to assume this can be used in the SRH calculation.

Since we haven't calculated any intrinsic values, the new code uses an alternate way of calculating n_t and p_t[18]:

$$n_t = N_C \exp\left(\frac{E_t - E_c}{kT}\right)$$
$$p_t = N_V \exp\left(\frac{E_v - E_t}{kT}\right)$$

Both the old and new code modifies the expressions by substituting carrier concentrations for the intermediate band when it is appropriate as shown in (3.1) and (3.2).

The last big change to the code is the way the correct fermi level split is found in part 2 of the code. For each cell voltage V, which is just the difference in quasi fermi levels for the conduction and valence band, the program must find the correct value for the quasi fermi level of the IB that gives a net zero current into the IB.

The old code uses a simple binary search of V_{ic}. For each search it then iterates through all values of n_{ib} for the one that minimizes absolute flux into the IB. This is therefore a loop inside a binary search, making it slow. The binary search terminates when the net flux fall beneath a relative limit.

The new code searches over V_{shift} instead, and directly calculates the n_{ib} that is needed, instead of trying many to find the best possible. The binary search has been improved as well. A binary search algorithm requires that it is known if the current try is too high or too low. Since a test of one particular V_{shift} can be invalid because of the errors in table 3.1, a regular binary search algorithm can't determine the next value to test. The modified algorithm uses information about how the error occur and uses them to set upper and lower boundaries for the search. Once both a lower and upper boundary is known, the algorithm turns into a conventional binary search. Figure 3.3 shows how the search looks.

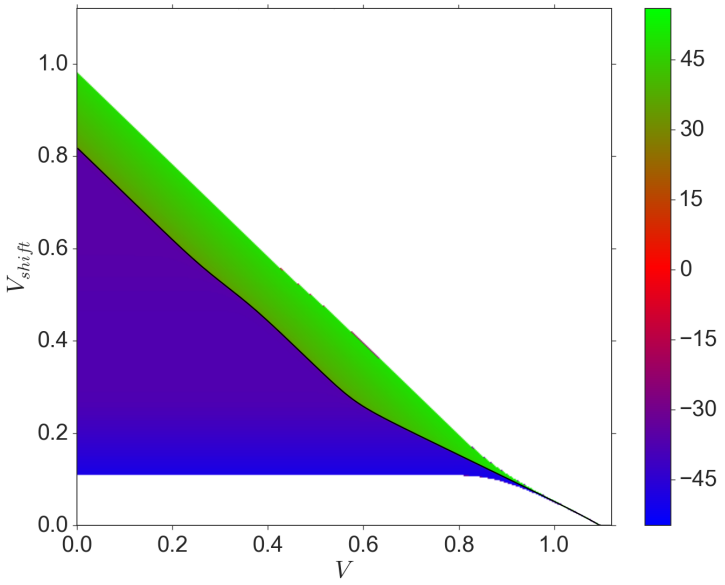

Figure 3.3: The figure shows a 2D map of the net flux into the intermediate band. The values on the right are the natural log of the flux in carriers per cm^3 per second. The black line between the green and red shows the final values of V_{shift}. The white space indicates that errors occur at those combinations of V and V_{shift}.

In contrast with the old code that stops after the net flux falls beneath a set relative value, the new code iterates a set number of times, `N_iter2` before terminating the search.

3.2 Photo- and electroluminescence

Luminescence analysis is the measurement and interpretation of light emitted when excited electron-hole pairs in a semiconductor are allowed to relax and recombine.

Luminescence as a phenomenon is seen many places in nature, e.g. bioluminescence in fireflies. It is also used in technology around us, with LEDs and cathode ray tubes being examples.

The method used to excite the electron-hole pairs determines the name of the luminescence. Photoluminescence is excited with light, cathodoluminescence with high energy electrons, and

electroluminescence is excited by an electric field driving a current through the material.

The results obtained from photoluminescence and electroluminescence are generally similar. A pure semiconductor will give a luminescence emission corresponding to the band gap, as the electrons in the conduction band relax to the valence band, emitting a photon.

There are differences, however. Electroluminescence requires electrical contacts to be present on the sample, and pure semiconductors are not suited for the technique since current won't flow easily through intrinsic materials.

A more interesting difference, when analysing IBSC, is the fact that EL might give a better representation of the spectrum around the IB. As argued by ref. [22], while PL could overemphasize the band gap peak due to much of the light being absorbed in the emitters, in electroluminescence the greater part of the recombination happens in regions with short lifetimes and in the depletion region, giving a better view of the IB region.

3.2.1 The luminescence equipment

The photoluminescence apparatus used for this project is delivered by Horiba. A sketch of the machine is provided in Figure 3.4. It works by selecting a single wavelength from a lamp using a monochromator. The reflected and emitted light is then guided with parabolic mirrors into the sample chamber where it is incident on the sample. The light from the sample is then passed through another monochromator into one of three detectors.

Instead of using the attached lamp, a 325 nm HeCd Laser from Kimmon Koha has been installed with mirrors to guide the light into the sample chamber. Lasers with other wavelengths are planned to be added in the future.

A setup for measuring electroluminescence was added by me last year, and I have during this work also made a holder for use in EL measurements.

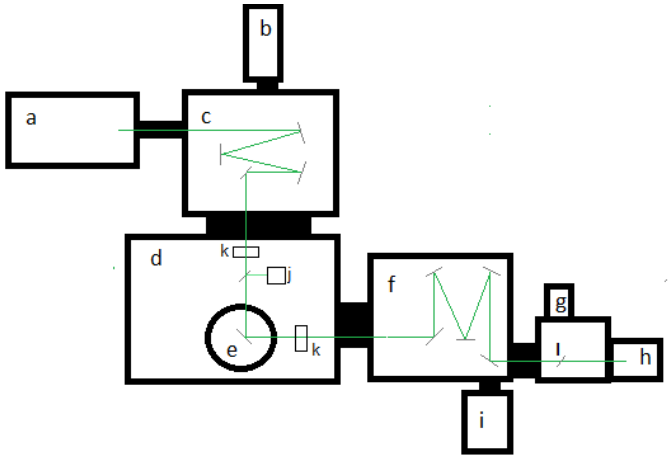

Figure 3.4: Sketch of the luminescence machine with the light path indicated in green. (a) Xenon lamp, (b) tungsten lamp for IR, (c) excitation monochromator, (d) box with filter holders, reference detector and sample chamber, (e) sample chamber, (f) emission monochromator, (g) PMT detector, (h) InGaAs detector, (i) Second InGaAs detector, (j) reference detector, (k) filter holders, (l) flip mirror. Figure taken from [23].

I have additionally used two more luminescence apparatuses with similar structure, to double-check the results obtained.

3.2.2 Stray light characterization

It has become apparent while doing PL analysis that the monochromators are not as efficient at selecting a single wavelength as we need for out applications. As well as the primary wavelength peak, we also see peaks from higher order refractions, as well as a broad band lower intensity spectrum from the lamp.

The higher order refraction peaks have a well known origin, and can be predicted in advance and be stopped by optical filters. The broad band light is harder to isolate away due to the overlap with the luminescence spectrum. It was also not obvious where this light was coming from, but I have considered two candidates.

The first would be diffuse reflection on the walls within the monochromator eventually finding its way into the sample chamber.

This seems like a reasonable hypothesis since the walls, although painted black, visibly reflects the light from the Xenon lamp.

The second is imperfections with the grating, reflecting a portion of the incoming light in addition to refracting the wanted order. This possibility was investigated because I observed a scratch on one of the diffraction gratings. Diffraction gratings will also invariably scatter a small amount of light. [24]

To test these hypotheses I created three spatial filters to selectively block light inside the monochromator. These filters were made out of cardboard, with the expectation of making something better if they proved effective. A picture of the filters, as well as their respective placement within the monochromator can be seen in figure 3.5.

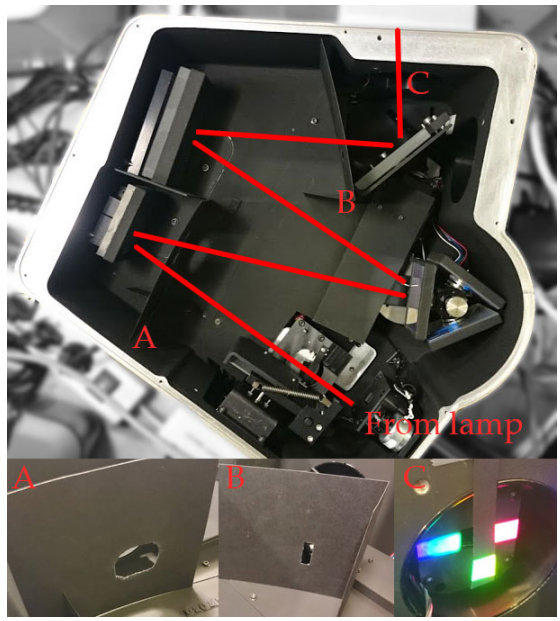

Figure 3.5: Position of the filters in the excitation monochromator. The three images show each filter. The light path is indicated in red.

Filter A is placed in front of the first collimating mirror, and restricts the light exiting the mirror, so that the light that would otherwise miss the refraction grating is absorbed. It also blocks light that would otherwise miss the exit into the sample chamber because it hit too low or too high on the exit. It should be noted that the

position of filter A right in front of the lamp, combined with the black color of the cardboard made it very hot, and the cardboard experienced some warping due to the heat.

Filter B is placed right in front of the last mirror before the exit into the sample chamber. It was designed to prevent indirect light from entering the sample chamber, blocking all light except a narrow part of the main light path. It also has a 'flap' on the side to add to this shading effect.

Filter C is placed right in front of the exit into the sample chamber. While filter A and B are used to see if stray light is due to diffuse reflection on the monochromator walls, filter C is used to test if the source of the stray light is in the main light path, indicating that the grating itself is the source. Since the light intensity in the main light path is very high, a layer of aluminum foil was later added to filter C to stop light transmitting through the cardboard.

The method for measuring the stray light is simple. You sweep both the excitation and the emission monochromator over the interesting wavelengths with no sample in the sample chamber. To ensure that all the light from the excitation side would be transmitted to the emission side, I used a mirror in place of the sample. Although mirror do not have a completely flat response curve, and might exhibit photoluminescence effects [25], it was good enough for my need.

A complication arose when trying to compare the magnitude of the stray light with and without the filters. The filters don't just block the unwanted parts of the light, but also reduces the overall intensity of light to enter the sample chamber. Therefore, to make a comparison with the setup without the filters I would have to choose a control point where I set the excitation and emission monochromators and measure the intensity. For each measurement I had to do this control measurement with no filters, with only filter A, and with both A and B filters. Then I would divide all the measured samples by the control point.

Since I expected stray light to be blocked by the filters, this reference point could not be a point with stray light. I usually chose a higher order refraction peak as control point, such as exciting at 400 nm and measuring the 'emission' at 800 nm. Ideally I would like to use

Table 3.2: These are the different measurements I did when measuring stray light. The first number is the blaze wavelength of the excitation monochromator, the second number is the the blaze wavelength of the emission monochromator. All values are in units nanometer.

Grating blaze wavelength	Wavelength range	Reference point	Filter
330/500	250–330/340–600	400/800	None, A,
330/1000	550–790/800–1100	500/1000	None, A,
1000/1000	550–790/800–1100	555/1110	None, A,
1000/1000	650–1500/800–1500	-	C

the main peak, i.e. the intensity when both monochromators are set at the same wavelength. However this intensity is too great, and will oversaturate the detectors. A reference point was not chosen in the measurement with filter C, as it is not intended as a viable way of reducing stray light, but only to determine the source.

Because of the way the reference diode works, the data from the photo-detectors are uncorrected. The reference diode is supposed to correct for temporal changes in the lamp output, but sums intensity over all wavelengths. To display the data, a logarithmic scale is used to show the signal to reference ratio in decibel, defined as

$$S_{\mathrm{dB}} = 10 \ln \frac{S}{R}$$

where S is the measured intensity and R is the intensity at the chosen reference point. For measurements with filter C, the signal is simply shown on a logarithmic axis.

To characterize the stray light in the monochromators, I measured three different ranges of wavelengths with different combinations of diffraction gratings, as listed in table 3.2. Each measurement was repeated done with no filters, the A filter and both A and B filters.

Chapter 4

Results and Discussion

This chapter will present the result obtained from the band filling simulator and the stray light characterization. After the results are presented, they are examined in the discussion part.

4.1 Band filling simulation

4.1.1 Comparison with old code

Before I present new results, I believe it is sensible to compare the outputs of the new program with the old one. Apart from the change in SRH recombination calculation, the calculations are the same; but the search heuristics might introduce other differences. Comparisons have been with the old code, with and without SRH recombination. Figure 4.1 compares the calculated carrier concentrations n, p and n_{ib} in the old and new script in the case of no SRH recombination. Figure 4.2 compares calculated SRH recombination in the old and new code. Comparisons of the other output values have not been plotted as they are nearly identical.

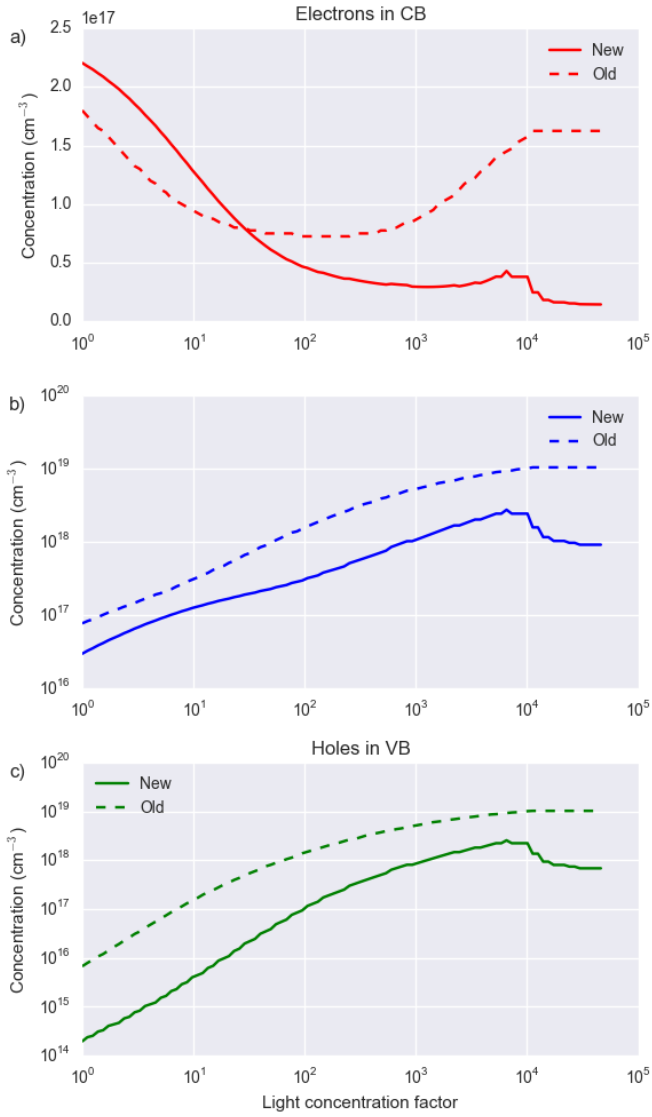

Figure 4.1: Comparison of the excited carrier concentrations in the a) intermediate, b) conduction, and c) valence band between the new and old code.

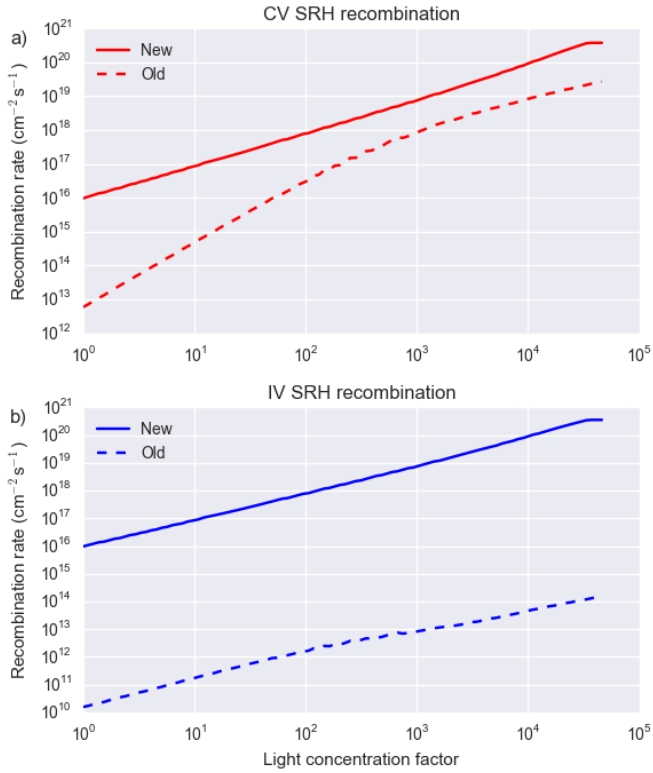

Figure 4.2: Comparison of the SRH recombination between the new and old code. The recombination is a) from the conduction band to the valence band and b) from the intermediate band to the valence band.

4.1.2 Comparison with SCAPS

To check the validity of the program output, the output was compared with an established program for solar cell simulations, SCAPS[1]. The comparison of the efficiency, conduction band electrons, valence band holes, max power point voltage, and generation and recombination rates are shown in figures 4.3, 4.4, 4.5, 4.6, and 4.7 respectively. SCAPS is not intended for intermediate band solar cells, so the input to my program has very few IB states to emulate a conventional solar cell. The output of the old program has also been included for completeness sake.

[1]SCAPS website: `http://scaps.elis.ugent.be/`

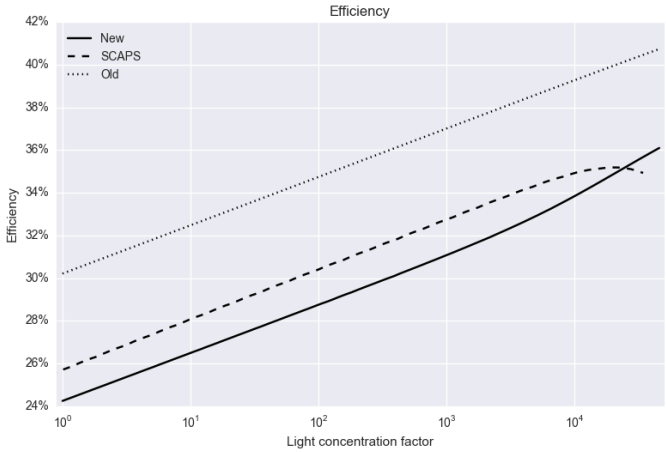

Figure 4.3: Comparison of the calculated efficiency between the new code, old code and SCAPS.

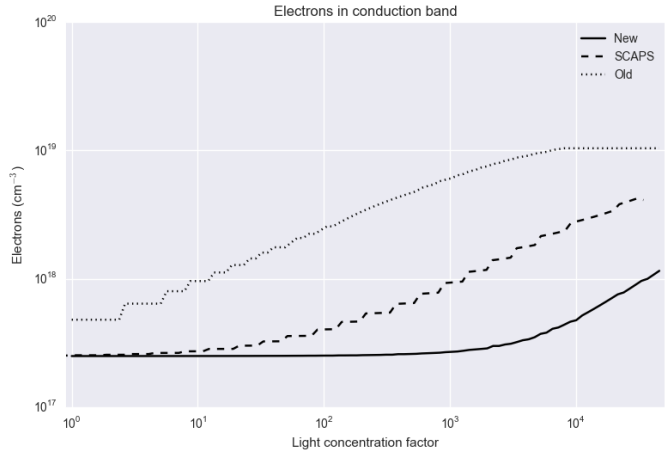

Figure 4.4: Comparison of the calculated electron concentration in the conduction band between the new code, old code and SCAPS.

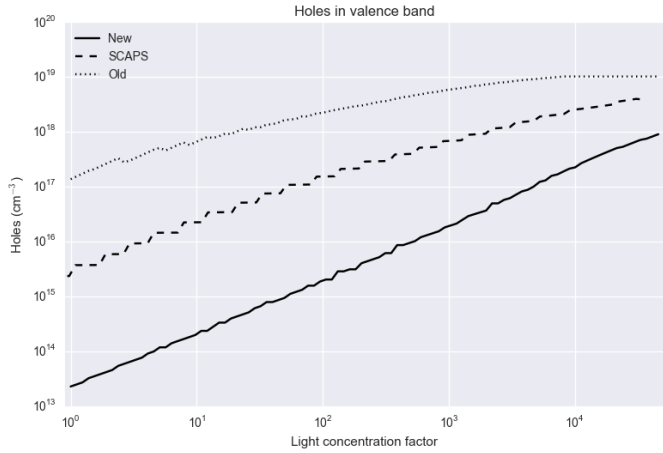

Figure 4.5: Comparison of the calculated hole concentration in the valence band between the new code, old code and SCAPS.

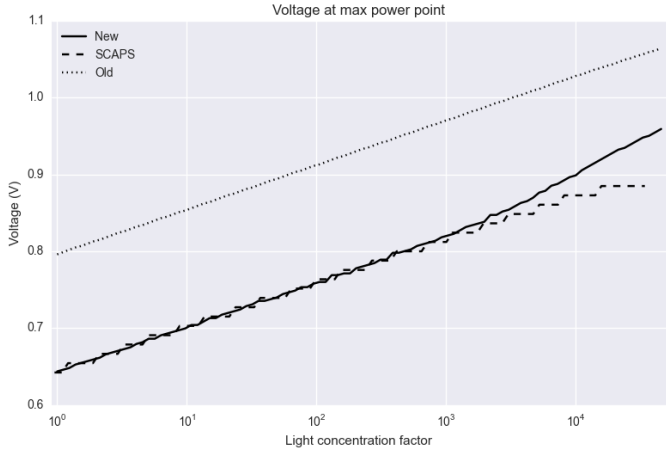

Figure 4.6: Comparison of the cell voltage at max power point between the new code, old code and SCAPS.

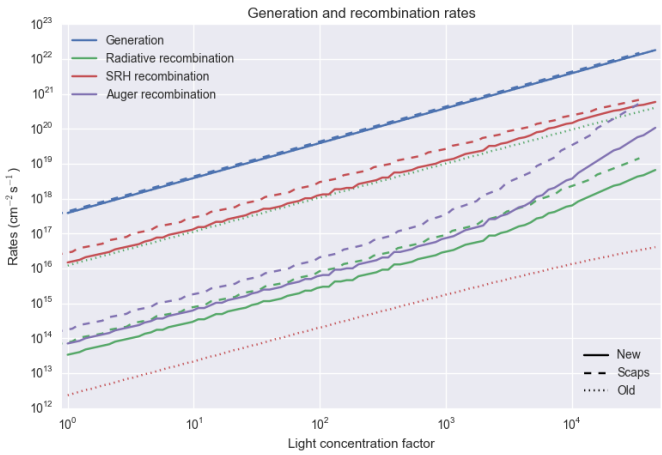

Figure 4.7: Comparison of generation and recombination rates between the new code, old code and SCAPS.

Effect of Auger recombination

To check the effects if the addition of Auger recombination, simulations were three different inputs; one with all Auger coefficients equal to $1 \times 10^{-31}\,\mathrm{cm}^{-6}\,\mathrm{s}^{-1}$, one with reduced Auger coefficient for the processes involving the intermediate band (T_1–T_4 in figure 3.2), and a control case without Auger recombination. The results are shown in figures 4.8, 4.9, 4.10 and 4.11.

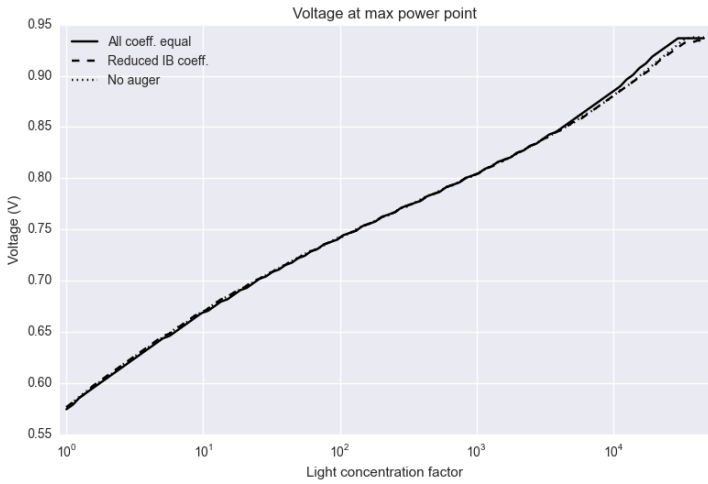

Figure 4.8: Comparison of cell voltage with different Auger coefficients.

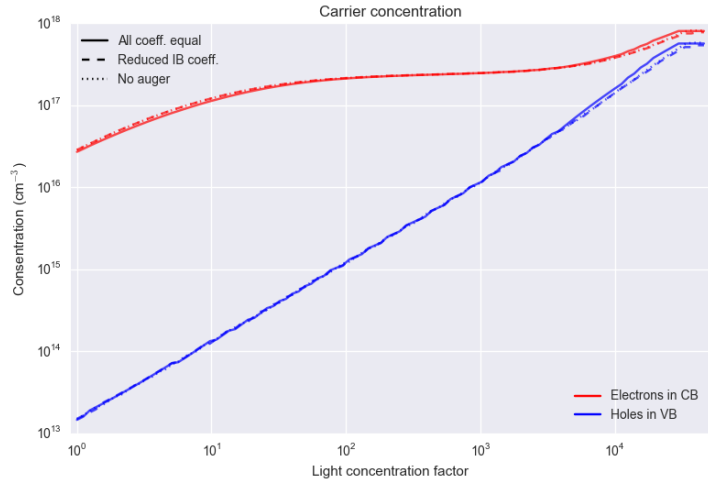

Figure 4.9: Comparison of carrier concentration in the CB and VB between cases with different Auger coefficients.

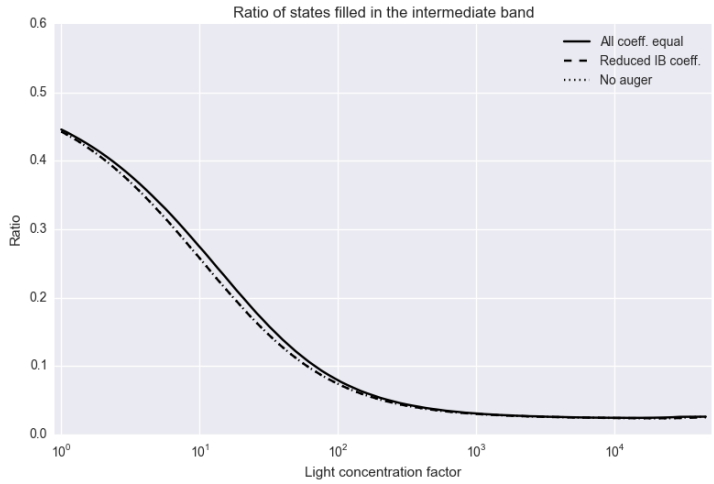

Figure 4.10: Comparison of IB filling between cases with different Auger coefficients. The lines plotted are the ration n_{ib}/N_{ib}.

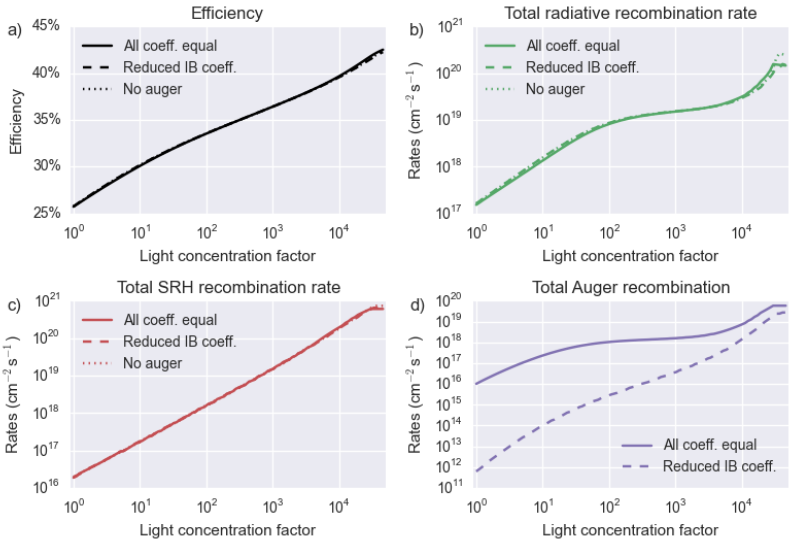

Figure 4.11: Comparison between cases with different Auger coefficients of a) Efficiency, b) Radiative recombination rate, c) SRH recombination rate and d) total auger recombination rate.

4.1.3 Discussion

For the comparison between the old an new program in figure 4.1 there is very little SRH recombination in the C case, so there shouldn't be much difference between the two programs. And for all variables other than the carriers, that is the case. The programs give the same answer for efficiency, peak voltage, as well as recombination and generation currents. This similarity hold except at high light intensities, where the two results diverge. The discrepancy in carrier concentrations is most likely explained by the new algorithm choosing at different n_{ib} to balance the carrier flux into the IB. Even though the difference looks large, the outputs are always within an order of magnitude of each other except at these high concentration factors. At high intensities, the result from new program becomes rough. This is probably due to the increased difficulty of finding the zero point at higher illuminations, as seen in figure 3.3. At high illuminations, very few values for V_{shift} are actually valid, so the program can not use the more efficient type of search. This is where the old program actually gets a better result. This can be fixed by running the new program for longer by setting `N_iter2` higher. The problem arises from the fact that on either side of the correct answer for V_{shift}, the value for n_{ib} varies a great deal, causing a chain reaction that changes the other values as well. To illustrate this, figure 4.12 shows what the calculated flux into the IB is at the end of the simulation for normal (x1) and fully concentrated (x46050) light for all voltages. At higher voltages and illumination, the value varies more, and it takes longer to find the root. Although the values seem high, they have to be seen in relation to the generation and recombination currents, which in this case are $1 \times 10^{14}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$.

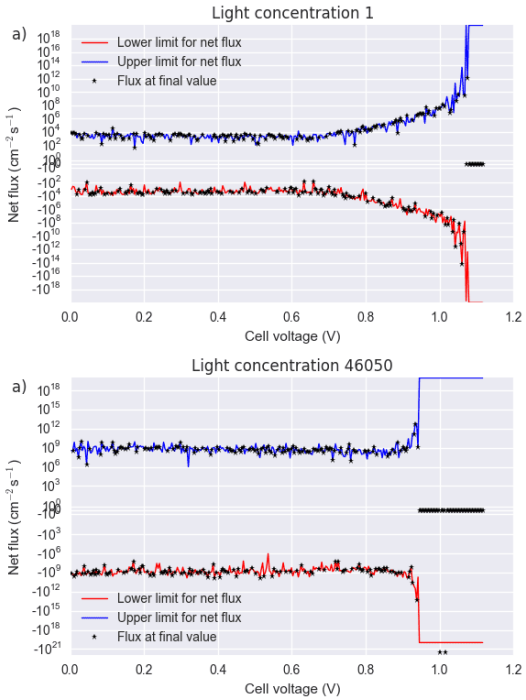

Figure 4.12: The values and minimum value envelopes for the calculated net flux of charge into the intermediate band. Values above a) 1.1 V and b) 0.97 V do not have stable solutions, and count as errors.

The comparison with SCAPS is problematic. My program and SCAPS are made to do different tasks. SCAPS simulates an complete solar cell with a p- and an n-type semiconductors and contacts under a realistic spectrum as opposed to a single material under a black body spectrum. In the comparison, I have chosen values for n and p from the n-type side of the silicon solar cell, where the material closely matches that of the input file with few IB states. Of course even within the same material the conditions change as you go from the surface of the cell to near the pn-junction, so I chose a spot near the middle as a compromise. For the rest of the values, the output from SCAPS counts overall values, from both p- and n-type regions, as well as recombination at the surfaces. There is probably also increased SRH recombination at the surfaces of the cell that my program does not simulate.

Another problem with the comparison with SCAPS is that my program can not simulate cells without an intermediate band. The

input file used has very few IB states, ($N_{ib} = 10 \times 10^5\,\mathrm{cm}^{-3}$), but still calculates a lot of values under the assumption that flux in and out of the intermediate band must be balanced.

As with the comparison with the old program, most values are within an order of magnitude of each other. The exception is the carrier concentrations, which is calculated to be much lower in my simulation. SRH recombination is much closer to SCAPS than the old program, but still bit below. The other recombinations are also close to the values calculated by SCAPS, and it is satisfying to see them following the same general pattern, as opposed to the straight lines of the old program.

For radiative recombination, SCAPS use a simple linear relationship, $R_{rad} = Bnp$, while my program simulates the solar cell as a black body. With such different approaches, so it is very nice to see that the results are close to each other. As with Auger recombination, which SCAPS seem to be modeling with $R_{auger} = C_h p^2 n + C_e n^2 p$, radiative recombination has an upwards slope at higher light intensity. This is due to a corresponding upwards slope of n and p, as shown in figures 4.4 and 4.5. The values for n and p for the SCAPS results are as mentioned just from a small slice of the n-type region, and might not be representable for the entire cells. Even in the calculation of n and p, two widely different models are used: As mentioned, my program calculates the carrier concentrations from the Fermi levels that balance the flux out and in of the intermediate band. The few states present in the IB might be part of the reason why the results for n and p are different.

Perhaps most troubling is the comparison of the efficiency in figure 4.3. The result from SCAPS begins to decrease at higher light concentrations. This is a result that makes intuitive sense as voltage is bounded by the band gap, and at some point recombination mechanisms will overcome the generation rate. That voltage is limiting can be seen in figure 4.6 where the SCAPS simulation seems to be hitting a limit. A similar limit exist in my program since at higher voltages it is harder to balance the IB charge balance. This can be seen in figure 3.3 where only a tiny strip of allowed combinations exist. For the comparison with SCAPS, however, this check for errors have been turned off, since getting the inequality $0 \leq n_{ib} \leq N_{ib}$ is very challenging with such a low N_{ib}. It is likely

that for even higher light concentrations, my program would also start seeing lower efficiencies as the voltage reaches the band gap. This result is troubling however, in that we would like to see if Auger recombination would put an efficiency limit on IBSCs, and we don't see an efficiency limit without an IB.

Adding Auger recombination to the simulation doesn't actually have much of an effect. A reduction in one recombination process causes an increase in another. This is because the intermediate band needs to be in balance no matter what. At low illuminations, the results are as expected. The added Auger recombination reduces the number of carriers in the VB and CB, which is directly tied in with reduced cell voltage, as seen in fig. 4.8. The radiative recombination rate is mainly dependent on this voltage, so we see a corresponding decrease in radiative recombination in fig. 4.11b. This reduced radiative and SRH recombination is counteracted by the added Auger recombination, so the efficiency remains more or less the same. It is interesting to note that the total recombination rate across all the recombination processes remains approximately the same. At higher illuminations we again see interesting behavior from the program, as the voltage is actually higher with added Auger recombination. This could possibly be explained if the added recombination gives a more favorable solution to the charge conservation formula for the IB. Auger recombination will also by necessity be relatively high in the IB processes, as they include either the term n_{ib} or p_{ib}. Since $p_{ib} = N_{ib} - n_{ib}$, one of these values will usually be large, causing the corresponding Auger recombination rate to be high.

4.2 Stray light characterization

The results from the 330/500 gratings are seen in figures 4.13 (No filters), 4.14 (Filter A) and 4.15 (Filter A and B). The results from the 330/1000 and 1000/1000 measurements show generally the same trends, with the 330/1000 having in average more stray light. The results from the 330/1000 measurements can be found in figures 4.16 (No filters), 4.17 (Filter A) and 4.18 (Filter A and B). For completion's sake, the results from the 1000/1000 gratings are attached in appendix C.1.

The figures are 2-dimensional plots of the intensity measured by the detector with varying excitation and emission wavelength. The x-axis of the figures indicates the wavelength which the detector is set to measure. The name "Emission wavelength" stems from the fact that the monochromator used is called the emission monochromator, used to select wavelengths of the emission light. The y-axis of the figures is the wavelength that the excitation monochromator is set to. This is the wavelength of light that is sent into the sample chamber.

The first line graph above the heat map shows the minimum value at that emission wavelength, or maximum in the case of filter C. The top line graph shows a reference spectrum of the Xenon lamp, obtained last semester during my project work. This reference spectrum was unfortunately not measured for the entire wavelength range being used here.

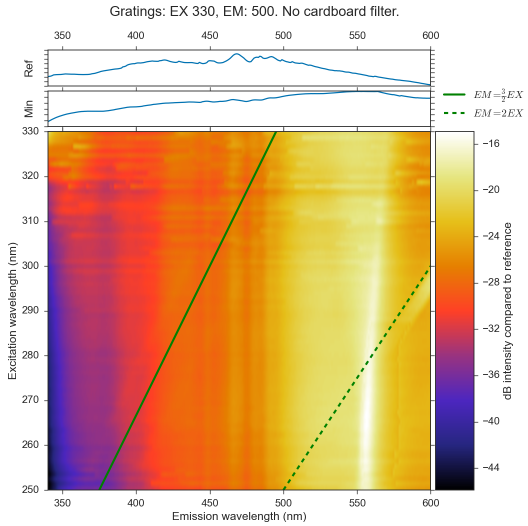

Figure 4.13: Stray light measurement for the 330 excitation grating and 500 emission grating with no filters.

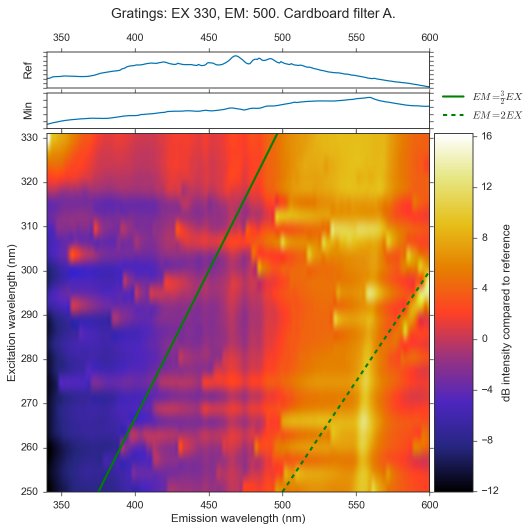

Figure 4.14: Stray light measurement for the 330 excitation grating and 500 emission grating with filter A.

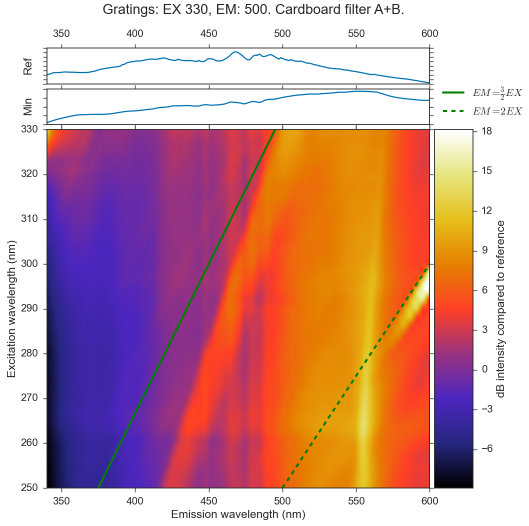

Figure 4.15: Stray light measurement for the 330 excitation grating and 500 emission grating with both filters A and B.

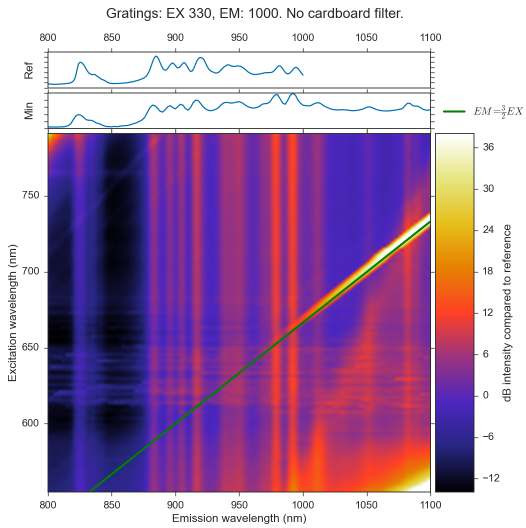

Figure 4.16: Stray light measurement for the 330 excitation grating and 1000 emission grating with no filters.

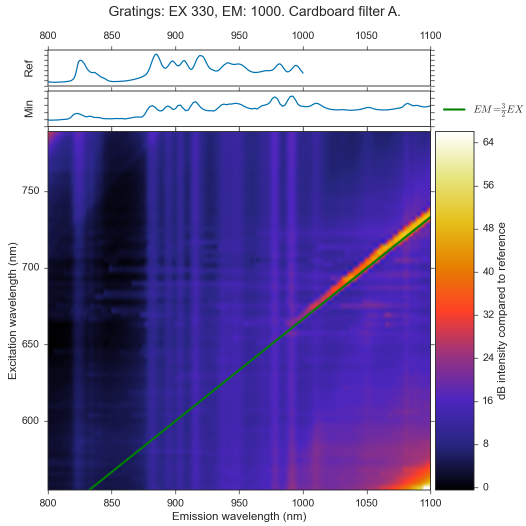

Figure 4.17: Stray light measurement for the 330 excitation grating and 1000 emission grating with filter A.

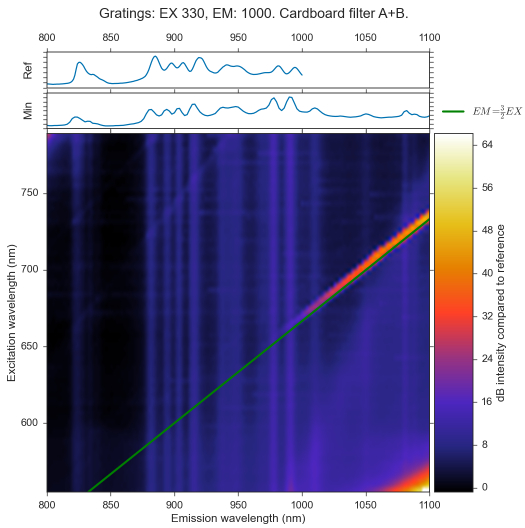

Figure 4.18: Stray light measurement for the 330 excitation grating and 1000 emission grating with filter A and B.

The measurement for the metal-backed filter C is shown in figure 4.19. The result without the metal backing is shown in figure C.4 in the appendix.

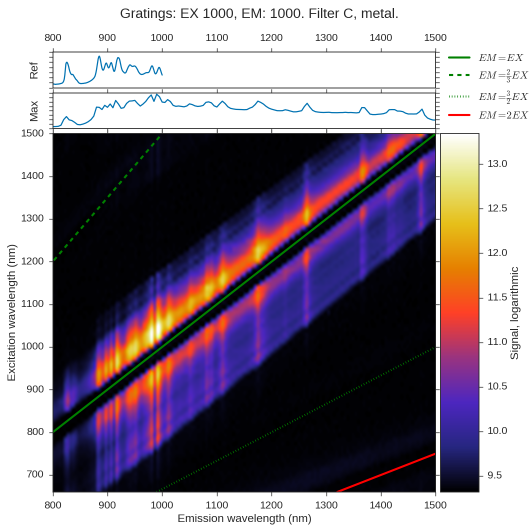

Figure 4.19: Stray light measurement for the 1000 excitation grating and 1000 emission grating with filter C with a metal backing. Compared to the filter without metal, there is reduced light in the $EM = EX$ line and the other higher order refractions.

4.2.1 Discussion

Ideally, these plots should be completely black, with only the line $EM = EX$ visible. Vertical lines on the graphs indicate stray light, and don't depend much on the excitation monochromator wavelength. That is, this light escapes from the excitation monochromator no matter what wavelength the monochromator is set to. Diagonal lines, some of which are indicated with lines on the graphs, originates from higher order refractions. Higher order refractions happen when the peridoic structure of the grating reflects two wavelengths of light at the same angle. This is especially obvious in C.3, where we can clearly see the second order line, $EM = 2EX$, where 500 nm light enters the emission monochromator where the second order refraction registers the light as 1000 nm. The $2EM = 3EX$ line likely originates from the same phenomena, but is a bit harder to explain.

Horizontal lines stems from temporal variations in the lamp intensity, as the data was measured in lines from the bottom left to the top right. The same is true for the bright spots that are

especially evident in figure 4.14, where the lamp intensity has varied greatly over time. This variation over time can be seen in figure 4.20, where I have simply measured the lamp intensity with the reference detector over time.

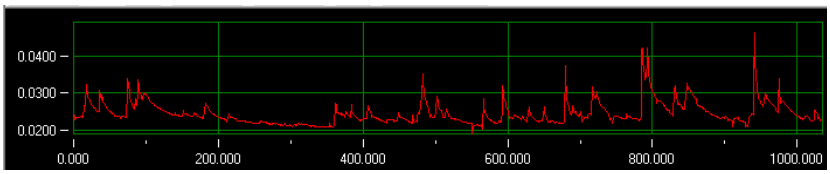

Figure 4.20: Measurement of the temporal variations in the lamp light with the reference detector. Horizontal axis is in units of seconds.

The addition of filter A decreases the amount of stray light, but does not eliminate it altogether. The addition of the B filter further reduces the stray light a bit. This seems to indicate that the stray light originates from scattered light from the walls in the monochromator. The results from measurements with the C filters does not seem to make as big of a difference, although it is hard to make a direct comparison as it is impossible to measure the direct light without overloading the sensor. This further indicates that the stray light does not originate from the main light path as it persists even when the main light path is blocked.

A further point to make is that the quality of the filters is not very good. The edges where the apertures were cut out are frayed and might cause undesirable effects. I would suggest, however, that the results obtained are good enough to make a conclusion.

Chapter 5

Conclusion

The focus for this master thesis was on simulating an intermediate band gap material with multiple recombination processes, and to analyze the stray light present in the luminescence equipment used by the IBSC group. A detailed explanation of the simulation program was given, as well as examples of input and output. The addition of Auger recombination did not seem to limit on the efficiency of the IBSC material under increasing illumination, but it contributes more than SRH recombination in the lowering of the efficiency.

To verify the results from the simulations, comparisons with SCAPS, an already established single junction solar cell simulator, were made. Although the program is not intended to simulate complete cells with pn-junctions, nor materials without an intermediate band, it performed closer to SCAPS than the old program on which it was based. The biggest difference was in the Shockley-Read-Hall recombination rate, where the program gives much lower values than SCAPS. SCAPS also exhibits a peak in efficiency when measuring over different illumination concentrations. This peak is not seen in my program. This may be because higher rates of recombination happens in other parts of the cell, like the pn-junction and contacts, which is not modeled by my program.

Addition of Auger recombination to the program does not seem to give an efficiency peak at higher illuminations. Rather, it seems like the Auger recombination rate is of the same order of magnitude as the other recombination processes, and that most of the loss in

efficiency is due to a small number of carriers lost to recombination processes causing a reduction of cell current.

To analyze the stray light in the luminescence equipment, optical filters were fashioned and placed in the excitation monochromator in different places and measuring how the light sent through the equipment responded. From my investigations, it is reasonable to believe that the stray light originates in scattered light on the walls in the monochromator. The walls are painted black, but are visibly scattering light.

Chapter 6

Further Work

There are three clear avenues for further work based on my thesis.

The first is to further improve on my program. There are many things to improve on, be it accuracy, efficiency or even usability. I have already talked about how SRH recombination is calculated to have much lower values than in other simulating software, and the main loop in part 2 could be modified to break after reaching a given precision. One thing that was planned but had to be dropped was the inclusion of a real solar spectrum, supplied by external files. The program is currently emulating solar radiation as a black body.

The second is to use my program on other materials. Silicon as a host material for IBSC is far from an optimal choice, but I didn't have time to investigate many different materials and material parameters.

Lastly is the continuation of work with luminescence. Since I started my work, a new laser has been installed. I hope that the laser, coupled with the information I have gathered about stray light will make work easier for the next people who will work on photoluminescence. During my work I also created a holder for doing electroluminescence, which I hope someone will use to do comparisons of PL and EL spectra as I was supposed to.

Bibliography

[1] Key World Energy Statistics 2014. International Energy Agency; OECD, 2014.

[2] Oliver Morton. Solar energy: A new day dawning?: Silicon valley sunrise. Nature, 443(7107):19–22, 2006. 10.1038/443019a.

[3] M. Green. Photovoltaics: coming of age. In Photovoltaic Specialists Conference, 1990., Conference Record of the Twenty First IEEE, pages 1–8 vol.1, May 1990.

[4] Christian Breyer and Alexander Gerlach. Global overview on grid-parity. Progress in Photovoltaics: Research and Applications, 21(1):121–136, 2013.

[5] Martin A. Green, Keith Emery, Yoshihiro Hishikawa, Wilhelm Warta, and Ewan D. Dunlop. Solar cell efficiency tables (version 45). Progress in Photovoltaics: Research and Applications, 23(1):1–9, 2015.

[6] Martin A. Green. Third generation photovoltaics: Ultra-high conversion efficiency at low cost. Progress in Photovoltaics: Research and Applications, 9(2):123–135, 2001.

[7] William Shockley and Hans J. Queisser. Detailed balance limit of efficiency of p-n junction solar cells. Journal of Applied Physics, 32(3), 1961.

[8] Jenny Nelson. The Physics of Solar Cells. Series on Properties of Semiconductor Materials. Imperial College Press, London, 2003.

[9] Martin A Green. Third generation photovoltaics: solar cells for 2020 and beyond. Physica E: Low-dimensional Systems and Nanostructures, 14(1–2):65 – 70, 2002.

[10] B. Van Zeghbroeck. Principles of semiconductor devices. `http://ecee.colorado.edu/~bart/book/book/title.htm`, 2011. [Online; accessed 2015-12-01; html].

[11] Peter T. Landsberg. Recombination in semiconductors. Cambridge University Press, Cambridge, 1991.

[12] Antonio Luque, Antonio Marti, and Colin Stanley. Understanding intermediate-band solar cells. Nat Photon, 6(3):146–152, 2012. 10.1038/nphoton.2012.1.

[13] Sedsel Fretheim Thomassen. InAs/(Al)GaAs quantum dots for intermediate band solar cells. PhD thesis, Norwegian University of Science and Technology, Department of Physics, 2012.

[14] Mohammadreza Nematollahi. Cr-doped ZnS for Intermediate Band Solar Cells. PhD thesis, NTNU, 10 2014.

[15] R. Kudrawiec, A. V. Luce, M. Gladysiewicz, M. Ting, Y. J. Kuang, C. W. Tu, O. D. Dubon, K. M. Yu, and W. Walukiewicz. Electronic band structure of $\mathrm{gan}_x\mathrm{p}_y\mathrm{as}_{1-x-y}$ highly mismatched alloys: Suitability for intermediate-band solar cells. Phys. Rev. Applied, 1:034007, Apr 2014.

[16] K. Yoshida, Y. Okada, and N. Sano. Effects of absorption spectra overlapping on structural design of intermediate band solar cells. In Photovoltaic Specialists Conference (PVSC), 2011 37th IEEE, pages 002117–002119, June 2011.

[17] Rune Strandberg and Turid Worren Reenaas. Photofilling of intermediate bands. Journal of Applied Physics, 105(12), 2009.

[18] Martin A. Green. Solar cells : operating principles, technology and system applications. University of New South Wales, Kensington, 1992.

[19] W.G.J.H.M. van Sark, L. Korte, and F. Roca. Physics and Technology of Amorphous-Crystalline Heterostructure Silicon Solar Cells. Engineering Materials. Springer Berlin Heidelberg, 2011.

[20] Alexis Vossier, Baruch Hirsch, and Jeffrey M. Gordon. Is auger recombination the ultimate performance limiter in concentrator solar cells? Applied Physics Letters, 97(19), 2010.

[21] Maryam Gholami Mayani and Turid Worren Reenaas. Shockley-read-hall recombination in pre-filled and photo-filled intermediate band solar cells. Applied Physics Letters, 105(7), 2014.

[22] I. Ramiro, E. Antolín, P.G. Linares, E. Hernández, A. Martí, A. Luque, C.D. Farmer, and C.R. Stanley. Application of photoluminescence and electroluminescence techniques to the characterization of intermediate band solar cells. Energy Procedia, 10(0):117 – 121, 2011. European Materials Research Society Conference. Symp. Advanced Inorganic Materials and Concepts for Photovoltaics.

[23] Ragnhild Strand. Photoluminescence of InAs/Ga(N)As quantum dot intermediate band solar cells. Master's thesis, NTNU, Høgskoleringen 1, 7491 Trondheim, 2014.

[24] J.M. Lerner and A. Thevenon. Diffraction gratings ruled & holographic. `http://www.horiba.com/in/scientific/products/optics-tutorial/diffraction-gratings/`. [Online; accessed 2015-10-06; website].

[25] Anatoly N. Trukhin, Krishjanis Smits, Georg Chikvaidze, Tatiana I. Dyuzheva, and Ludmila M. Lityagina. Luminescence of silicon dioxide — silica glass, α-quartz and stishovite. Central European Journal of Physics, 9(4):1106–1113, 2011.

[26] Fabiana Villela da Motta, Ana Paula de Azevedo Marques, Vinícius Dantas de Araújo, Mara Tatiane de Souza Tavares, Mauricio Roberto Bomio Delmonte, Carlos Alberto Paskocimas, Máximo Siu Li, Rubens Maribondo do Nascimento, and Elson Longo. Optical characterization of europium-doped indium hydroxide nanocubes obtained by Microwave-Assisted Hydrothermal method. Materials Research, 17:933 – 939, 08 2014.

Appendices

Appendix A

Attempt at luminescence spectroscopy

In this section I will explain why the section about luminescence spectroscopy was moved to the appendix. I will present the timeline of events as well as the partial results from the measurements done.

A.1 An introductory timeline

The apparatus that was meant to be used for luminescence spectroscopy is the one presented in the main text, chapter 3.2. Due to the problems with stray light, it was decided that I should spend the beginning of my thesis analyzing the stray light for a possible solution, as well as continuing on some work done previously on simulation of IBSC materials.

A decision was made between me and my supervisor to delay photoluminescence studies until a laser excitation source could be acquired, so my focus during the time was on the simulation program. In the period before the laser was installed, I also made two holders for use in EL measurements. The laser and its installation was delayed longer than expected, and I was only able to do my first measurements in the second week of August; two weeks before my initial deadline. After this I also encountered what appeared to be a software problem which left me unable to

do measurements on the apparatus. I was offered help by Professor Mikael Lindgren, who had a luminescence apparatus with a similar setup. The measurements done with this apparatus were done in the two last weeks of September, with one week left in my extended deadline.

The results obtained did not seem to be reproducible from day to day, and varied a great deal with small changes to sample position and orientation, even to the point where clear luminescence peaks would disappear from one measurement to another. As a last resort, I used the laser from the original machine to set up a makeshift apparatus with a handheld spectrometer as a sensor.

I approached Mikael Lindgren with the results, and we agreed that the quality of the results combined with the short period of time left were sufficient grounds to diminish the focus of the thesis on luminescence. This in turn meant that time had to be spent rewriting the thesis and expanding on the parts pertaining to the simulation program.

A.2 Apparatus

The three apparatuses used were the one presented in the main text, chapter 3.2; a similar setup with a flash-lamp as excitation light and a photomultiplier detector; and a handheld spectrometer from Avantes with the laser from the first apparatus as excitation light.

Electroluminescence measurements were only performed with the first apparatus, and used a sample holder made for the purpose of this thesis, as well as the EL setup I developed in my project work before the start of this thesis. A picture of the sample holder, as well as a prototype, is shown in figure A.1. The reason why we wanted to do both photoluminescence and electroluminescence on the same samples were that previous studies on IBSC show that EL might be better at giving information about the IB materials, as opposed to the host material. [22]

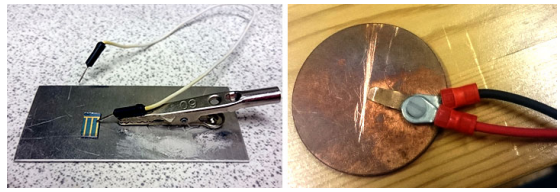

Figure A.1: The sample holders used for electroluminescence. The picture to the left is a prototype with a back contact of aluminum, and the picture to the right is the final version made of copper and wires attached to both front and back contact.

A.3 Samples

The samples that were provided to do spectroscopy on were zinc sulfide (ZnS) solar cells doped with chromium (Cr). The samples were made by Mohammadreza Nematollahi using pulsed laser deposition and molecular beam epitaxy. A list of the samples are presented in table A.1. The cells are deep level intermediate solar cells, where a high density of dopants form a deep level in the bandgap of the host material [14].

Table A.1: The growth method and composition of the measured samples. Samples without chromium are reference samples. PLD stands for Pulsed Laser Deposition, and MBE stands for Molecular Beam Epitaxy.

Sample name	Growth method	Sample composition
140203	PLD	ZnS (Reference)
140219	PLD	Cr:ZnS
140204	PLD	Cr:ZnS
140131	PLD	Cr:ZnS
140221	PLD	Cr:ZnS
140220	PLD	Cr:ZnS
130816-03	MBE	ZnS (Reference)
130816-04	MBE	ZnS (Reference)
121025	MBE	Cr:ZnS
121105	MBE	Cr:ZnS
121122	MBE	Cr:ZnS
140314	MBE	Cr:ZnS
140207	MBE	Cr:ZnS

A.4 Results

Many measurements were taken, but most resulted in noise only. The best results obtained are presented for sample 140203(PLD) in figure A.2, sample 140219(PLD) in figure A.3, sample 130816-03(MBE) in figure A.4 and sample 130816-04(MBE) in figure A.5. Measurement parameters change between measurements. The most common change is moving the sample in the sample chamber; but excitation wavelength are in some cases different.

The reason why only these samples have good measurements, is mainly due to the fact that they are the samples I focused on. When I started the measurements I wanted to have a reference spectrum, so I started with measuring the reference cells. Other cells were tested, but as mentioned resulted in pure noise. In addition, all attempts at obtaining an EL spectrum, for reference samples 140203(PLD) and 130816-03(MBE), resulted in only noise.

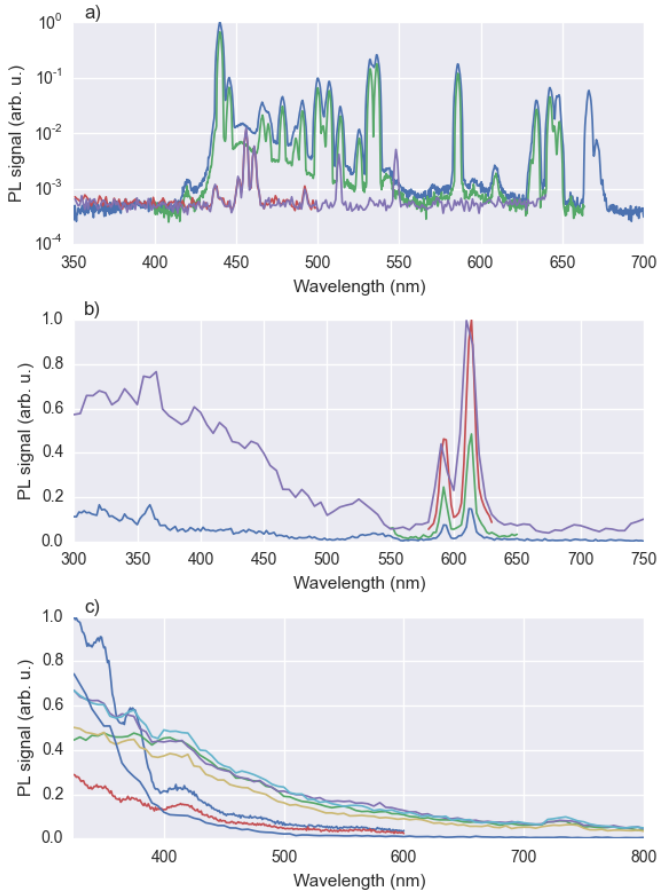

Figure A.2: This figure shows the results from the reference PLD sample, which was the sample I did most measurements on. Here are presented a sample of the best results from three days. The top graph (a) shows the results from four measurements using the original apparatus on a logarithmic scale. The middle graph (b) shows the results from four measurements using the replacement apparatus, each with a distinct double peak around 600 nm. The bottom graph (c) shows seven measurement result obtained with the same setup the following week with a bump at 410 nm and a small bump at around 730 nm.

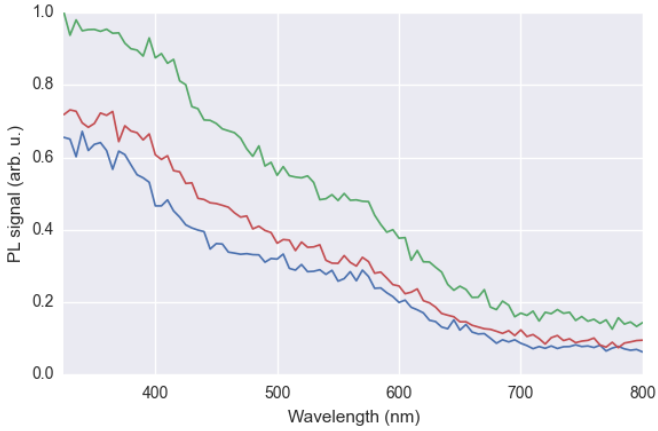

Figure A.3: This figure shows the results from three measurements on sample 140219(PLD). It was the only non-reference sample I got any decent signal from. This result is from the replacement apparatus.

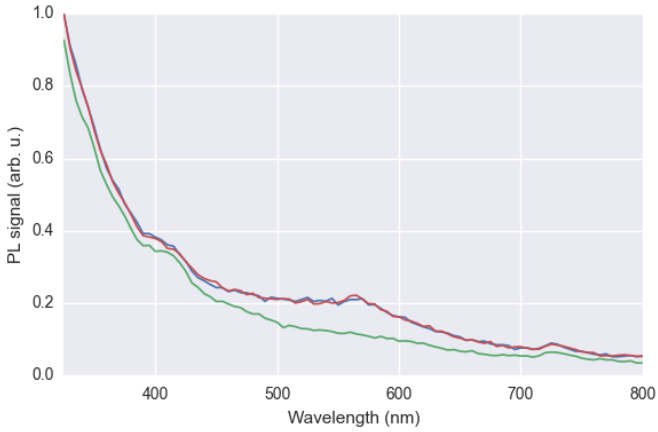

Figure A.4: This figure shows the results from measurements on the reference sample 130816-03(MBE). There is a small bump after 400 nm and at 725 nm. There is also a broader bulge around 550 nm for two of the measurements. This result is from the replacement apparatus.

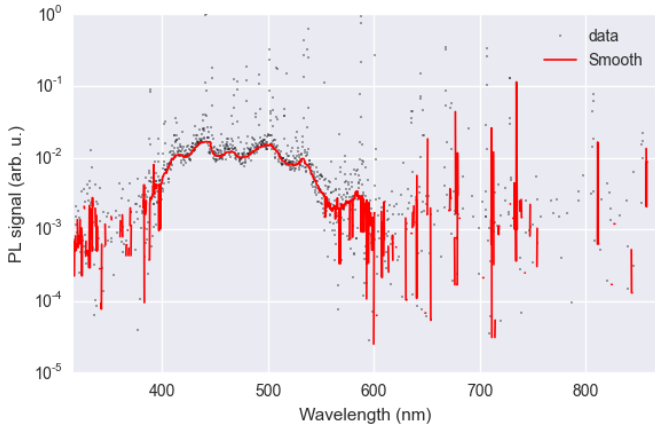

Figure A.5: This figure shows the results from measurements on the reference sample 130816-04(MBE). This measurement was taken with the handheld spectrometer which has a lot of noise. The logarithmically scaled signal measure shows a broad flat peak from 425 nm to 525 nm.

A.5 Discussion and Conclusion

The results obtained from measurements on sample 140203 are both the best and most puzzling of the results obtained. The measurements on the original apparatus consists of almost purely sharp peaks in two different patterns for the four measurements, as shown in fig. A.2(a). The measurements with highest amplitude seems to have a downward sloping peak from 450 nm to 550 nm. The excitation light in this case was a laser, but I can think of no other explanation for this result than saying that this signal is from the laser with peak wavelength 325 nm. The reason why we don't see the laser at lower wavelengths is because I used a long pass filter to block laser light below 450 nm, which seems to match the beginning of the slope. It is still weird that laser light has this shape.

After gaining access to the second luminescence apparatus, the first measurements done was the ones seen in figure A.2(b). These seem really good, with a clear double peak on each side of 600 nm. This is very reminiscent of what is seen in photoluminescence spectra of europium doped indium hydroxide [26], but I was unable to find any other results with similar peaks in ZnS.

Repeating the previous experiment yielded a completely different result, as seen in fig. A.2(c). The double peak is gone, and we are left with a sharp slope down from what I assume is light from the lamp. The result with slightly less intensity was done with the excitation monochromator set 5 nm lower to see if the small peaks at 400 nm and 725 nm would move with it as higher order refractions would, but they seem to be in the same position. The broader peak in the middle, around 550 nm disappeared, which could possibly be because the sample was rotated in the sample chamber between the measurements.

Moving on to sample 140219, which is the only sample with chromium that I god any signal from, we see a gradual slope without any obvious features in fig. A.3. This leads me to believe what I have measured is simply the reflected light from the excitation lamp.

The result for sample 130816-04 was included here to show what the signal from the handheld monochromator looked like. The signals were very noisy, but this sample showed a consistent broad and flat peak from 425 nm to 525 nm. The signal was very weak, but this could possibly be signal from the band gap edge, which usually is at around 350 nm for pure ZnS. The addition of chromium could possibly have decreased the band gap sufficiently for this to be an explanation.

These results are just the best of many measurements I've done, but serve to show that PL results were either very bad, with noise and lamplight dominating over any possible luminescence, or inconsistent, as in the double peak of sample 140203 disappearing. The only result I feel is repeatable is the two small peaks at 400 nm and 725 nm, which show up in the measurements of two different reference samples. Those results were measured on the same apparatus however, so the peaks might be characteristic to the lamp.

In conclusion, I would very much have liked to do more measurements on different samples and apparatuses, but lining up samples and getting a good signal is a very time consuming and frustrating process, and I was under heavy time constraints. I realize in hindsight that I should have done more measurements on the original apparatus, but noise problems in the lab subconsciously led

me to focus on the replacement apparatus.

The sample solar cells used could also be the reason for the bad result. It would have been nice to try other samples, but since the plan for the thesis was to compare EL and PL, I would need complete solar cells with contacts, of which there were very few to choose from.

Appendix B

Simulation program

B.1 Source code

B.1.1 main.py

```
#!/usr/bin/env python3

# This program was written by Peter Kristoffersen
# (petek@stud.ntnu.no) as part of his master's thesis.
# It is based on previous programming work by
# Maryam Gholami Mayani and Turid Worren Reenaas which
# is in turn based on theoretical work by Rune Strandberg
#
# Although the program was written for python 3.x,
# it will run with minor changes in python 2, specifically
# changing the names of some unicode variable identifiers.

import numpy as np
import scipy.integrate
import time
import datetime as dt


class C:  # Constants
    q = 1.602176487e-19  # Elementary charge [C]
    k = 8.6173324e-5  # Boltzmann's constant [eV/K]
    h = 4.135667516e-15  # Planck constant [eV s]
    c = 2.99792e8  # Speed of light [m/s]
    pi = np.pi
    k_strange = 1.38065E-19  # Boltzmann constant in weird units [kg cm^2/(s^2 K)]
    h_strange = 6.62607E-30  # Planck constant [kg cm^2/s]


class Error:
    err_negative_current = 2 ** 0  # 1
    err_n_ib_negative = 2 ** 1  # 2
    err_n_ib_too_large = 2 ** 2  # 4
    err_mu_vi_too_large = 2 ** 3  # 8
    err_mu_ic_too_large = 2 ** 4  # 16


class Result:
    def __init__(self):
        for k in ("ib_net_flux", "E_0",
                  "G_vc", "G_vi", "G_ic",
                  "R_vc", "R_vi", "R_ic",
                  "SRH_vc", "SRH_vi", "SRH_ic",
                  "auger_vc", "auger_vi", "auger_ic",
```

```
                    "n", "p", "n_t", "n_ib",
                    "F_v", "F_c", "F_t", "F_ib",
                    "V", "V_shift", "P", "η", "J", "errors"):
            setattr(self, k, None)


class Parameters:
    def __init__(self, **kwargs):
        for k, v in kwargs.items():
            setattr(self, k, v)

        def m_from_N(N, T):
            """
            Calculates effective mass from carrier concentration
            :param N: carrier concentration in cm^-3
            :param T: cell temperature in K
            return effective mass in kg
            """
            return (N / 2) ** (2 / 3) * C.h_strange ** 2 / (2 * C.pi * C.k_strange * T)

        if not hasattr(self, "m_e"):
            self.m_e = m_from_N(self.N_c, self.T)
        if not hasattr(self, "p_e"):
            self.m_p = m_from_N(self.N_v, self.T)
        if not hasattr(self, "E_vi"):
            self.E_vi = self.E_g - self.E_ic
        if not hasattr(self, "E_ic"):
            self.E_ic = self.E_g - self.E_vi
        if hasattr(self, "auger_C"):
            if not hasattr(self, "auger_C_T1"):
                self.auger_C_T1 = self.auger_C
            if not hasattr(self, "auger_C_T2"):
                self.auger_C_T2 = self.auger_C
            if not hasattr(self, "auger_C_T3"):
                self.auger_C_T3 = self.auger_C
            if not hasattr(self, "auger_C_T4"):
                self.auger_C_T4 = self.auger_C
            if not hasattr(self, "auger_C_B1"):
                self.auger_C_B1 = self.auger_C
            if not hasattr(self, "auger_C_B2"):
                self.auger_C_B2 = self.auger_C

    # Default simulation parameters, can be overridden
    T = 300  # Room temperature [K]
    T_s = 6000  # Temperature of sun [K]
    sun_x = 1  # Light concentration [1]
    max_integration = 18  # [eV], sets the upper limit for black body integration. Set to np.inf only if needed

    # Default program parameters, can be overwritten
    N_iter = 50  # Search iterations for equilibrium values
    N_iter2 = 20  # Search iterations for detailed balance
    V_num = 100  # Number of voltages to calculate
    calculate_auger = True
    calculate_SRH = True

    @classmethod
    def from_import(cls, import_name):
        import importlib.machinery
        loader = importlib.machinery.SourceFileLoader('input_file', import_name)
        s = loader.load_module()
        init_dict = dict((k, v) for k, v in s.__dict__.items() if not k.startswith("__"))
        return cls(**init_dict)

    @classmethod
    def from_dict(cls, dictionary):
        return cls(**dictionary)


class Experiment:  # Base class for experiments
    def __init__(self, parameters):
        self.parameters = parameters
        self.results = Result()

    def calculate_SRH(self, N0, N1, Et, n, n0, p, p0, Fn, Fp):
        P = self.parameters
        B = 1 / (C.k * P.T)  # eV^-1
        v_n = np.sqrt(3 * C.k_strange * P.T / P.m_e)  # cm/s
        v_p = np.sqrt(3 * C.k_strange * P.T / P.m_p)  # cm/s
        τ_n = 1 / (v_n * P.σ_n * P.N_t)  # seconds
        τ_p = 1 / (v_p * P.σ_p * P.N_t)  # seconds
        nt = N1 * np.exp((Et - Fn) * B)  # cm^-3
        pt = N0 * np.exp((Fp - Et) * B)  # cm^-3
```

```
126             R = (n * p - n0 * p0) / (τ_p * (n + nt) + τ_n * (p + pt))  # cm^-3 s^-1
127             return R * P.w  # cm^-2 s^-1
128 
129         def calculate(self):
130             ## Set up shortcuts
131             P = self.parameters
132             R = self.results
133             B = 1 / (C.k * P.T)
134             B_s = 1 / (C.k * P.T_s)
135 
136             ## PART 1:
137             ## Calculate equilibrium filling and energies using poisson eq.
138             ##
139             ## We use binary search of the poisson equation to find poisson(E_0)=0
140             ## In most (all) cases, poisson(E_0) will be monotonically increasing.
141             ## We rely on this fact.
142 
143             def poisson(E_0):
144                 p0 = P.N_v / (np.exp(E_0 * B))  # holes in VB with E_i
145                 n0 = P.N_c / (np.exp((P.E_g - E_0) * B))  # electrons in CB with E_i
146                 n0_ib = P.N_ib / (1 + np.exp((P.E_vi - E_0) * B))  # ib states filled
147                 n0_t = P.N_t / (1 + np.exp((P.E_t - E_0) * B))  # trap states filled
148                 charge = n0 + n0_ib + n0_t + P.N_a - p0 - P.N_d
149                 return charge, p0, n0, n0_ib, n0_t
150 
151             # This method can fail if N_v is very large compared to the other N
152             # Then, poisson(E_i) does not monotonically increase
153             # Then there might also be two zeros.... I think
154             b_min, b_max = 0, P.E_g
155             R.E_0 = (b_max + b_min) / 2
156             for _ in range(P.N_iter):  # This is the binary search
157                 if poisson(R.E_0)[0] > 0:
158                     b_max = R.E_0
159                 else:
160                     b_min = R.E_0
161                 R.E_0 = (b_max + b_min) / 2
162             p0, n0, n0_ib, n0_t = poisson(R.E_0)[1:]
163             p0_ib = P.N_ib - n0_ib
164 
165             ##
166             ## PART 2:
167             ## We solve the continuity equation.
168             ## For steady state, the net charge flow into the IB must be 0.
169             ## For each fermi level split, V, we find the Vshift, the distance
170             ## from the VB edge to the VB fermi level that fixes net IB current to 0.
171             R.V = np.linspace(0, P.E_g, P.V_num + 2)[1:-1]  # Skip first and last
172 
173             # Set up constants and values that is used in the calculation:
174             θ_c = np.arcsin(1 / P.nr)  # Critical angle for internal reflection
175             cons_frac = P.sun_x / 46050  # Concentration fraction
176             # Integrals for detailed balance current production
177             s = 1E-4 * (2 * P.nr ** 2) / (C.h ** 3 * C.c ** 2)  # Prefactor for integral [cm^-2 eV^-3 s^-1]
178 
179             def DB_integral(E_low, E_high, μ, B):  # Black body flux integral
180                 """Calculates the flux of photons with energy in the range
181                    E_low to E_high from a black body with chemical potential
182                    μ and temperature B = 1/kT
183                    μ is an array of chemical potentials."""
184 
185                 def f(x, μ):
186                     return x ** 2 / (np.exp((x - μ) * B) - 1)
187 
188                 return s * np.array([scipy.integrate.quad(f, E_low, E_high, (m,))[0] for m in μ])  # [cm^-2 s^-1]
189 
190             def DB_integral2(critical_angle, α):  # Internal reflection integral
191                 """Calculates the integral for internal reflection.
192                    :param critical_angle: Critical angle
193                    :param a: array (!) of absorption coefficients
194                    Returns an array (!) of the results"""
195 
196                 def f(x, a):
197                     cs = np.cos(x)
198                     sn = np.sin(x)
199                     return (1 - np.exp(-2 * P.w * a / cs)) * cs * sn
200 
201                 return 2 * np.pi * np.array([scipy.integrate.quad(f, 0, critical_angle, (ai,))[0] for ai in α])
202 
203             if P.E_vi > P.E_ic:
204                 vi_flux_s = DB_integral(P.E_vi, P.E_g, np.zeros(1), B_s)[0]  # Solar photons in vi energy range
205                 ic_flux_s = DB_integral(P.E_ic, P.E_vi, np.zeros(1), B_s)[0]  # Solar photons in ic energy range
206             else:  # Eic is larger than Evi
207                 vi_flux_s = DB_integral(P.E_vi, P.E_ic, np.zeros(1), B_s)[0]  # Solar photons in vi energy range
```

```
            ic_flux_s = DB_integral(P.E_ic, P.E_g, np.zeros(1), B_s)[0]  # Solar photons in ic energy range
        vc_flux_s = DB_integral(P.E_g, P.max_integration, np.zeros(1), B_s)[0]

        def recombination(R):
            """Given an array of Vshift,
               calculates total flux of carriers into IB.
               Should be 0 for steady state.
               Calculates for all values of (V,Vshift)."""
            lenV = len(R.V)
            R.errors = np.zeros(lenV, dtype=np.int32)  # bitmask of errors
            # Calculate fermi levels
            R.F_v = R.Vshift
            R.F_c = R.Vshift + R.V
            R.F_t = R.F_v  # Simplified
            # Calculate carrier concentrations
            R.p = P.N_v / (np.exp(R.F_v * B) + 1)
            R.n = P.N_c / (np.exp((P.E_g - R.F_c) * B) + 1)
            R.n_t = P.N_t / (np.exp((P.E_t - R.F_t) * B) + 1)
            R.n_ib = R.p + P.N_d - R.n - R.n_t - P.N_a  # Since charge is conserved, rho_f = 0
            R.errors += (R.n_ib < 0) * Error.err_n_ib_negative  # Check that 0<n_ib<N_ib
            R.errors += (R.n_ib > P.N_ib) * Error.err_n_ib_too_large
            p_ib = P.N_ib - R.n_ib
            R.F_ib = P.E_vi - np.log(P.N_ib / R.n_ib - 1) / B
            # Recombination and generation
            μ_vi = R.F_ib - R.F_v
            μ_ic = R.F_c - R.F_ib
            R.errors += (μ_vi > P.E_vi) * Error.err_mu_vi_too_large
            R.errors += (μ_ic > P.E_ic) * Error.err_mu_ic_too_large
            vi_flux_a = np.zeros(lenV)
            ic_flux_a = np.zeros(lenV)
            int2_vi = np.zeros(lenV)
            int2_ic = np.zeros(lenV)
            # Absorption coefficients
            α_vi = P.σ_vi * p_ib
            α_ic = P.σ_ic * R.n_ib
            int2_vi[R.errors == 0] = DB_integral2(θ_c, α_vi[R.errors == 0])
            int2_ic[R.errors == 0] = DB_integral2(θ_c, α_ic[R.errors == 0])
            # Solar Generation
            R.G_vi = cons_frac * vi_flux_s * int2_vi
            R.G_ic = cons_frac * ic_flux_s * int2_ic
            if P.E_vi > P.E_ic:
                # Ambient photons in vi energy range
                vi_flux_a[R.errors == 0] = DB_integral(P.E_vi, P.E_g, μ_vi[R.errors == 0], B)
                # Ambient photons in ic energy range
                ic_flux_a[R.errors == 0] = DB_integral(P.E_ic, P.E_vi, μ_ic[R.errors == 0], B)
            else:
                # Ambient photons in vi energy range
                vi_flux_a[R.errors == 0] = DB_integral(P.E_vi, P.E_ic, μ_vi[R.errors == 0], B)
                # Ambient photons in ic energy range
                ic_flux_a[R.errors == 0] = DB_integral(P.E_ic, P.E_g, μ_ic[R.errors == 0], B)
            R.R_vi = vi_flux_a * int2_vi
            R.R_ic = ic_flux_a * int2_ic
            # Calculate SRH recombination
            R.SRH_vi = np.zeros(lenV)
            R.SRH_ic = np.zeros(lenV)
            if hasattr(P, 'calculate_SRH') and P.calculate_SRH is True:
                if P.E_t < P.E_vi:  # SRH from IB to VB via trap
                    R.SRH_vi = self.calculate_SRH(P.N_v, P.N_ib, P.E_t, R.n_ib, n0_ib, R.p, p0, R.F_ib, R.F_v)
                    R.SRH_ic = np.zeros(lenV)
                elif P.E_t > P.E_vi:  # SRH from CB to IB via trap
                    R.SRH_vi = np.zeros(lenV)
                    R.SRH_ic = self.calculate_SRH(P.N_ib, P.N_c, P.E_t, R.n, n0, p_ib, p0_ib, R.F_c, R.F_ib)

            # Calculate Auger recombination
            if hasattr(P, 'calculate_auger') and P.calculate_auger is False:
                R.auger_ic = np.zeros(lenV)
                R.auger_vi = np.zeros(lenV)
            else:
                R.auger_ic_e = P.auger_C_T1 * R.n * R.n * p_ib * P.w
                R.auger_ic_h = P.auger_C_T2 * R.n * R.p * p_ib * P.w
                R.auger_ic = R.auger_ic_e + R.auger_ic_h
                R.auger_vi_e = P.auger_C_T3 * R.p * R.n * R.n_ib * P.w
                R.auger_vi_h = P.auger_C_T4 * R.p * R.p * R.n_ib * P.w
                R.auger_vi = R.auger_vi_e + R.auger_vi_h
            R.auger_vi *= R.errors == 0
            R.auger_ic *= R.errors == 0
            net_auger = R.auger_ic - R.auger_vi  # Net carrier flow into IB
            net_SRH = R.SRH_ic - R.SRH_vi
            R.ib_net_flux = (R.G_vi - R.G_ic +
                             R.R_ic - R.R_vi +
                             net_auger + net_SRH)  # Net carrier flow into IB

```

```
290             def divisions(a, b):
291                 l = [a, b]
292                 i = 1
293                 while True:
294                     new = [n for n in np.linspace(a, b, 2 ** i + 1) if n not in l]
295                     yield from new
296                     l.extend(new)
297                     i += 1
298 
299             trials = dict()
300             upper_bound = np.zeros(P.V_num, dtype="bool")
301             lower_bound = np.zeros(P.V_num, dtype="bool")
302 
303             b_min, b_max = (np.zeros(P.V_num),
304                             P.E_g - R.V)
305 
306             R.Vshift = (b_max + b_min) / 2
307             indexes = np.arange(P.V_num)
308 
309             for counter in range(P.N_iter2):  # This is the binary search
310                 recombination(R)
311                 errors = R.errors != 0
312                 too_high_n_ib = (R.errors & Error.err_n_ib_too_large) != 0
313                 too_low_n_ib = (R.errors & Error.err_n_ib_negative) != 0
314                 negative = R.ib_net_flux < 0
315                 set_lower = too_high_n_ib | (negative & ~errors)
316                 set_upper = too_low_n_ib | (~negative & ~errors)
317                 upper_bound |= set_upper
318                 b_max[set_upper] = R.Vshift[set_upper]
319                 lower_bound |= set_lower
320                 b_min[set_lower] = R.Vshift[set_lower]
321                 updated = set_upper | set_lower
322 
323                 indeterminate = errors & ~too_high_n_ib & ~too_low_n_ib
324 
325                 # If no bounds are known, or if the current Vshift doesn't give enough info:
326                 no_bounds = (~upper_bound & ~lower_bound & ~updated) | indeterminate
327                 check_index = indexes[no_bounds]
328                 for k in list(trials.keys()):
329                     if k not in check_index:
330                         del trials[k]
331                 for j in check_index:
332                     if j not in trials:
333                         trials[j] = divisions(b_min[j], b_max[j])
334                     R.Vshift[j] = next(trials[j])
335 
336                 # Case no upper bound
337                 no_upper = ~upper_bound & lower_bound & ~updated
338                 b_max[no_upper] = (b_max[no_upper] + b_min[no_upper]) / 2
339                 # Case no lower bound
340                 no_lower = ~lower_bound & upper_bound & ~updated
341                 b_min[no_lower] = (b_max[no_lower] + b_min[no_lower]) / 2
342                 # Case both bounds
343 
344                 R.Vshift[~no_bounds] = ((b_max + b_min) / 2)[~no_bounds]
345 
346             del trials
347 
348             recombination(R)
349             ## PART 3:
350             ## Now that we have solved the continuity equation, we calculate
351             ## the efficiency.
352 
353             # Generation vc
354             int2_vc = DB_integral2(θ_c, np.ones(1) * P.α_vc)
355             R.G_vc = np.ones(P.V_num) * cons_frac * vc_flux_s * int2_vc[0]
356 
357             # Recombination vc
358             vc_flux_a = DB_integral(P.E_g, P.max_integration, R.V, B)  # Time consuming
359             R.R_vc = vc_flux_a * int2_vc
360 
361             # SRH vc
362             if hasattr(P, 'calculate_SRH') and P.calculate_SRH is True:
363                 R.SRH_vc = self.calculate_SRH(P.N_v, P.N_c, P.E_t, R.n, n0, R.p, p0, R.F_c, R.F_v)
364             else:
365                 R.SRH_vc = np.zeros(P.V_num)
366 
367             # Calculate Auger recombination
368             if hasattr(P, 'calculate_auger') and P.calculate_auger is True:
369                 R.auger_vc_e = P.auger_C_B1 * R.n * R.n * R.p * P.w
370                 R.auger_vc_h = P.auger_C_B2 * R.n * R.p * R.p * P.w
371                 R.auger_vc = R.auger_vc_h + R.auger_vc_e
```

```
372          else:
373              R.auger_vc = np.zeros(P.V_num)
374
375          # net losses from CB to VB
376          # Since flux into IB = flux out of IB at steady state,
377          # only include loss mechanisms from IB->VB
378          nonradiative_losses = R.SRH_vc + R.SRH_vi + R.auger_vc + R.auger_vi
379
380          R.J = C.q * (R.G_vc - R.R_vc +
381                       R.G_vi - R.R_vi -
382                       nonradiative_losses)
383          R.errors += (R.J < 0) * Error.err_negative_current
384
385          R.P = R.J * R.V  # W/cm^2
386
387          # efficiency
388          A33 = lambda E: E ** 3 / (np.exp(E * B_s) - 1)
389          B33 = scipy.integrate.quad(A33, 0, P.max_integration)[0]
390          P_sun = cons_frac * ((2 * C.pi) / (C.h ** 3 * C.c ** 2)) * B33  # Sun effect [eV/(m^2 s)]
391          P_sun_SI = P_sun * 1.602177E-19 * 1E-4  # Convert to J/(cm^2 s) (1596 W/m^2 for default values)
392          R.η = R.P / P_sun_SI
393
394          return "Done"
395
396
397  ### Code below is for parsing command line arguments
398
399  def input_to_dicts(param_file):
400      """Takes an input file and outputs a numpy matrix with a column
401         for each variable in the input file and calculated value"""
402      import itertools
403      from collections import defaultdict
404      import importlib.machinery
405      import sys
406      import copy
407
408      loader = importlib.machinery.SourceFileLoader('input_file', param_file)
409      loader.__cached__ = None
410      inp = loader.load_module()
411      imp_dict = dict((k, v) for k, v in inp.__dict__.items()
412                      if isinstance(v, (int, float, list, tuple, str, bool, dict)) and not k.startswith("__"))
413      common_input = dict((k, v) for k, v in imp_dict.items() if not hasattr(v, '__iter__'))
414      # All parameters that are not iterable are common to all calculations
415      multi_input = dict((k, sorted(v)) for k, v in imp_dict.items() if
416                         k not in common_input)  # All parameters that vary for each experiment
417      multi_keys = multi_input.keys()
418      multi_combs = list(
419          itertools.product(*[multi_input[k] for k in multi_keys]))  # Generate all combinations of parameters
420      multi_dict = [dict(zip(multi_keys, values)) for values in multi_combs]
421      input_columns = defaultdict(list)
422      output_columns = defaultdict(list)
423      input_dict = copy.deepcopy(multi_dict)
424      start_time = time.time()
425      for counter, input_variables in enumerate(input_dict):
426          input_variables.update(common_input)
427          parameters = Parameters.from_dict(input_variables)
428          ex = Experiment(parameters)
429          time_now = time.time()
430          time_elapsed = int(time_now - start_time)
431          if counter == 0:
432              time_total = 0
433          else:
434              time_total = (time_elapsed / counter) * len(input_dict)
435          time_left = int(time_total - time_elapsed)
436          print("Calculating case {}/{} of {}. (Time elapsed: {}. Time left: {})".format(counter + 1,
437                                                                                         len(input_dict),
438                                                                                         param_file,
439                                                                                         dt.timedelta(
440                                                                                             seconds=time_elapsed),
441                                                                                         dt.timedelta(seconds=time_left)))
442          ex.calculate()
443          no_outputs = len(ex.results.V)
444          for k, i in [(k, i) for k, i in ex.results.__dict__.items() if
445                       isinstance(i, np.ndarray) and len(i) == no_outputs]:
446              output_columns[k].extend(i)
447          for k, i in input_variables.items():
448              input_columns[k].extend([i] * no_outputs)
449      return input_columns, output_columns, common_input, multi_dict
450
451
452  def main(arguments=None):
453      import sys
```

```
    import os
    if arguments:
        args = parser.parse_args(arguments.split())
    else:
        args = parser.parse_args()
    if args.format is None:
        args.format = ['txt']
    for input_file in args.input_file:
        if not os.path.isfile(input_file):
            raise Exception("File {} does not exist".format(input_file))
    for input_file in args.input_file:
        print("Processing file {}.".format(input_file))
        inp_folder = os.path.split(input_file)[0]
        input_columns, output_columns, common, multi = input_to_dicts(input_file)
        [writers.writers[writer](input_file, input_columns,
                                 output_columns, common, multi) for writer in writers.writers if writer in args.format]


if __name__ == "__main__":
    import argparse
    import writers

    parser = argparse.ArgumentParser(description="Calculates conditions in a semiconductor under illumination.")
    parser.add_argument('input_file', nargs="+")
    parser.add_argument('-f', '--format', required=False, choices=writers.writers.keys(), nargs="*",
                        help="Specify the format of the output, if not specified, will output .txt file.")
    main()
```

B.1.2 writers.py

```
#!/usr/bin/env python3

# The functions defined in this file helps save the results from an Experiment
# class. More writers can be added by defining a function with the same call
# signature and adding it to the 'writers' dictionary at the bottom.
# writing xls and pd files require pandas installed.

import numpy as np
import os


def test_filename(root, extension):
    """Returns a valid filename that does not already exist of the form
       <root> (N).<extension>, where N is the smallest possible number"""
    import glob

    existing = glob.glob("{}*.{}".format(root, extension))
    n = 2
    current = "{}.{}".format(root, extension)
    while current in existing:
        current = "{} ({}).{}".format(root, n, extension)
        n += 1
    return current


def txt_writer(input_file, input_columns, output_columns, common_inputs, multi_inputs):
    def output_formatter(val):
        if isinstance(val, (str, int, np.integer)):
            return str(val)
        if isinstance(val, bool):
            return str(val)
        if isinstance(val, (float, np.float)):
            return str(val)
        if isinstance(val, list):
            return "[" + ", ".join(output_formatter(value) for value in val) + "]"
        if isinstance(val, np.ma.core.MaskedConstant):
            return "---"
        if isinstance(val, np.ma.MaskedArray):
            arr = ["---" if m else a for a, m in zip(val, np.ma.getmaskarray(val))]
            return output_formatter(arr)
        if isinstance(val, np.ndarray):
            return output_formatter(list(val))
        raise TypeError("No formatting rule for type {}.".format(str(type(val))))

    output_string = list()
    output_string.append("Output of {}".format(input_file))
    output_string.append("# Common parameters:")
```

```
48      for k, v in sorted(common_inputs.items()):
49          v = output_formatter(v)
50          output_string.append("# {:>15} = {}".format(k, v))
51      no_experiments = len(multi_inputs)
52      for i, mo in enumerate(multi_inputs):
53          output_string.append("\n\n## Calculation {}/{}".format(i + 1, no_experiments))
54          output_string.append("## Calculation specific variables:")
55          selected_rows = [True] * len(output_columns['V'])
56          for k, v in sorted(mo.items()):
57              for j, select in enumerate(selected_rows[:]):
58                  selected_rows[j] = selected_rows[j] and v == input_columns[k][j]
59              v = output_formatter(v)
60              output_string.append("## {:>15} = {}".format(k, v))
61          output_string.append("\n### Calculation results:")
62          for k, v in sorted(output_columns.items()):
63              v = [vv for sel, vv in zip(selected_rows, v) if sel]
64              v = output_formatter(v)
65              output_string.append("### {:>15} = {}".format(k, v))
66      output_name = os.path.splitext(input_file)[0] + "-output"
67      output_name = test_filename(output_name, "txt")
68      print("Writing output (.txt) file [{}]".format(output_name))
69      with open(output_name, mode="w", encoding="UTF-8") as f:
70          f.write("\n".join(output_string))
71
72
73  def dataframe_writer(input_file, input_columns, output_columns, common_inputs, multi_outputs):
74      import pandas
75      columns = dict()
76      columns.update(input_columns)
77      columns.update(output_columns)
78      headings = sorted(input_columns.keys()) + sorted(output_columns.keys())
79      df = pandas.DataFrame(columns, columns=headings)
80      return df
81
82
83  def pd_writer(input_file, input_columns, output_columns, common_inputs, multi_outputs):
84      df = dataframe_writer(input_file, input_columns, output_columns, common_inputs, multi_outputs)
85      output_name = os.path.splitext(input_file)[0] + "-output"
86      output_name = test_filename(output_name, "pd")
87      print("Writing pandas (.pd) pickle file [{}]".format(output_name))
88      df.to_pickle(output_name)
89
90
91  def xls_writer(input_file, input_columns, output_columns, common_inputs, multi_outputs):
92      df = dataframe_writer(input_file, input_columns, output_columns, common_inputs, multi_outputs)
93      output_name = os.path.splitext(input_file)[0] + "-output"
94      output_name = test_filename(output_name, "xls")
95      print("Writing excel (.xls) file [{}]".format(output_name))
96      df.to_excel(output_name)
97
98  def np_writer(input_file, input_columns, output_columns, common_inputs, multi_outputs):
99      import pickle
100     columns = dict()
101     columns.update(input_columns)
102     columns.update(output_columns)
103     keys = sorted(columns.keys())
104     types = [type(columns[k][0]).__name__ for k in keys]
105     types = ['unicode' if t == 'str' else t for t in types] # Workaround for unicode support of numpy
106     matrix = [columns[k] for k in keys]
107     dtypes = list(zip(keys, types))
108     np_array = np.array(list(zip(*matrix)), dtype=dtypes)
109     output_name = os.path.splitext(input_file)[0] + "-output"
110     output_name = test_filename(output_name, "np")
111     print("Writing numpy (.np) pickle file [{}]".format(output_name))
112     with open(output_name, 'wb') as f:
113         pickle.dump(np_array, f)
114
115
116 writers = {'txt': txt_writer,
117            'pd': pd_writer,
118            'np': np_writer,
119            'xls': xls_writer}
```

B.2 Sample inputs

This section contains some of the input files used for my simulations. The file in section B.2.1 has additional comments to clarify variables, as well as the rationale behind the values chosen.

B.2.1 Case A

Case A is the "default" input with SRH recombination between the IB and the VB. This file is commented with information about the different parameters, as well as some sources for the values.

```
import numpy as np
# Input files are regular python programs, so imports are allowed

# Simulation parameters
T = 300          # Room temperature [K]
T_s = 6000       # Temperature of sun [K]
sun_x = list(np.logspace(0, np.log10(46050), 100)) # Light consentration [dim. less]
# If a parameter is a list, all combination of possible parameters will be calculated

# Program parameters
N_iter = 50      # Search iterations for intrisic values
N_iter2 = 50     # Search iterations for detailed balance
V_num = 500      # Number of voltages to calculate
calculate_auger = True
calculate_SRH = True

## Material parameters
# Effective density of of states: [1/cm^3]
N_ib = 5e17      # Intermediate band     (2
N_d = N_ib/2     # Donors
N_a = 0          # Acceptors
N_t = 1e16       # Traps                 (5
N_v = 1.8e19     # Valence band          (1
N_c = 3.2e19     # Conduction band       (1
nr = 3.5         # Refractive index      (3

# Effective masses [kg]
# m_e (optional)
# m_p (optional)
# Effective masses are calculated automatically if not given explicitly

# Energies: [eV]
# You don't have to define E_ic and E_vi. One of them is enough.
E_g = 1.12       # Energy gap            (1
E_ic = 0.21      # Energy gap from IB to CB
#E_vi optional   # Energy gap from VB to IB
E_t = 0.5        # Trap energy level

# Absorption coefficients [1/cm]
α_vc = 1e4       # (2

# Absorption cross section  [cm^2]
σ_vi = 3e-14     # (2
σ_ic = σ_vi      # (2

# Capture cross section [cm^2]
σ_n = 1e-15      # (5
σ_p = 1e-17      # (5

# width [cm]
w = 5e-4         # (5

# Auger recombination coefficients: [cm^6/s]
# See figure in thesis, or fig 2.1.1 in
# "Recombination in Semiconductors" by P.T. Landsberg
# For explanation of these terms
# If supplied a number the program will use the same coefficient
```

```
# for all processes.
auger_C = 1E-31             # (4
# auger_C_T1 = auger_C
# auger_C_T2 = auger_C
# auger_C_T3 = auger_C
# auger_C_T4 = auger_C
# auger_C_B1 = auger_C
# auger_C_B2 = auger_C      # Done automatically


## Sources:
## 1: http://www.ioffe.ru/SVA/NSM/Semicond/Si/bandstr.html
## 2: Photofilling of intermediate bands: R. Strandberg & T. W. Reenaas
## 3: http://www.pmoptics.com/silicon.html
## 4: http://iopscience.iop.org/article/10.1088/0031-8949/8/4/007/meta
## 5: Shockley-Read-Hall recombination in pre-filled and photo-filled intermediate band solar cells: M. G. Mayani & T. W. Reenaas
```

B.2.2 Case C

Case C is similar to case A, but with no trap states.

```
import numpy as np
# Simulation parameters
sun_x = list(np.logspace(0, np.log10(46050), 100)) # Light consentration
N_iter2 = 300    # Search iterations for detailed balance
V_num = 300     # Number of voltages to calculate
calculate_auger = False
calculate_SRH = False
# Effective density of of states: [cm^-3]
N_ib = 5e17     # Intermediate band
N_d = N_ib/2    # Donors
N_a = 0         # Acceptors
N_t = 0         # Traps
N_v = 1.8e19    # Valence band
N_c = 3.2e19    # Conduction band
nr = 3.5        # Refractive index
# Energies: [eV]
E_g = 1.12      # Energy gap
E_ic = 0.21     # Energy gap from IB to CB
E_t = 0.5       # Trap energy level
# Absorption coefficients [cm^-1]
α_vc = 1e4
#Absorption cross section [cm^2]
σ_vi = 3e-14
σ_ic = σ_vi
# Capture cross section [cm^2]
σ_n = 1e-14
σ_p = 1e-17
# width [cm]
w = 5e-4
# Auger recombination coefficients: [cm^6/s]
auger_C = 1E-31
```

B.2.3 Case B

Similar to B, but with Auger recombination calculation disabled. Used in comparison with the old program which didn't support Auger recombination.

```
import numpy as np
# Simulation parameters
sun_x = list(np.logspace(0, np.log10(46050), 100)) # Light consentration
N_iter2 = 150    # Search iterations for detailed balance
V_num = 650     # Number of voltages to calculate
calculate_auger = False
calculate_SRH = True
# Effective density of of states: [cm^-3]
N_ib = 5e17     # Intermediate band
```

```
N_d = N_ib/2    # Donors
N_a = 0         # Acceptors
N_t = 1e16      # Traps
N_v = 1.8e19    # Valence band
N_c = 3.2e19    # Conduction band
nr = 3.5        # Refractive index
# Energies: [eV]
E_g = 1.12      # Energy gap
E_ic = 0.21     # Energy gap from IB to CB
E_t = 0.5       # Trap energy level
# Absorption coefficients [cm^-1]
α_vc = 1e4
#Absorption cross section [cm^2]
σ_vi = 3e-14
σ_ic = σ_vi
# Capture cross section [cm^2]
σ_n = 1e-14
σ_p = 1e-17
# width [cm]
w = 5e-4
```

B.2.4 Case A0

Similar to A, but with very few IB states, emulating a conventional solar cell material. Used in comparison with SCAPS.

```
import numpy as np
# Simulation parameters
sun_x = list(np.logspace(0, np.log10(46050), 100)) # Light consentration
N_iter2 = 50    # Search iterations for detailed balance
V_num = 500     # Number of voltages to calculate
calculate_auger = True
calculate_SRH = True
# Effective density of of states: [cm^-3]
N_ib = 1e5      # Intermediate band
N_d = 2.5e17    # Donors
N_a = 0         # Acceptors
N_t = 1e16      # Traps
N_v = 1.8e19    # Valence band
N_c = 3.2e19    # Conduction band
nr = 3.5        # Refractive index
# Energies: [eV]
E_g = 1.12      # Energy gap
E_ic = 0.21     # Energy gap from IB to CB
E_t = 0.5       # Trap energy level
# Absorption coefficients [cm^-1]
α_vc = 1e4
#Absorption cross section [cm^2]
σ_vi = 3e-14
σ_ic = σ_vi
# Capture cross section [cm^2]
σ_n = 1e-14
σ_p = 1e-17
# width [cm]
w = 5e-4
# Auger recombination coefficients: [cm^6/s]
auger_C = 1E-31
```

B.2.5 Case Ared

Similar to A, but with the Auger recombination coefficients corresponding to processes involving the IB reduced. This was done see the effect of different value of the Auger coefficients.

```
import numpy as np
# Simulation parameters
```

```
sun_x = list(np.logspace(0, np.log10(46050), 100)) # Light consentration
N_iter2 = 50     # Search iterations for detailed balance
V_num = 500      # Number of voltages to calculate
calculate_auger = True
calculate_SRH = True
# Effective density of of states: [cm^-3]
N_ib = 5e17      # Intermediate band
N_d = N_ib/2     # Donors
N_a = 0          # Acceptors
N_t = 1e16       # Traps
N_v = 1.8e19     # Valence band
N_c = 3.2e19     # Conduction band
nr = 3.5         # Refractive index
# Energies: [eV]
E_g = 1.12       # Energy gap
E_ic = 0.21      # Energy gap from IB to CB
E_t = 0.5        # Trap energy level
# Absorption coefficients [cm^-1]
α_vc = 1e4
#Absorption cross section [cm^2]
σ_vi = 3e-14
σ_ic = σ_vi
# Capture cross section [cm^2]
σ_n = 1e-14
σ_p = 1e-17
# width [cm]
w = 5e-4
# Auger recombination coefficients: [cm^6/s]
auger_C = 1E-31
auger_C_T1 = 1E-40
auger_C_T2 = 1E-40
auger_C_T3 = 1E-40
auger_C_T4 = 1E-40
```

B.3 Sample output

B.3.1 Case A Output (excerpt)

This is an excerpt from the text output when running input A, `main.py A.inp -f txt`. Energies are given in eV. Recombination and generation currents are given in $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$. Electric current density J is given in $\mathrm{A\,cm}^{-2}$. Power density P is given in $\mathrm{W\,cm}^{-2}$. Voltage is given in V. Efficiency, η is given as a ration of power in over power out.

```
Output of A.inp
# Common parameters:
#             E_g = 1.12
#            E_ic = 0.21
#             E_t = 0.5
#             N_a = 0
#             N_c = 3.2e+19
#             N_d = 2.5e+17
#            N_ib = 5e+17
#          N_iter = 50
#         N_iter2 = 50
#             N_t = 1e+17
#             N_v = 1.8e+19
#               T = 300
#             T_s = 6000
#           V_num = 500
#         auger_C = 1e-31
#   calculate_SRH = True
# calculate_auger = True
#              nr = 3.5
#               w = 5e-06
#           a_vc = 1000000.0
```

```
#            s_ic = 3e-18
#             s_n = 1e-19
#             s_p = 1e-21
#            s_vi = 3e-18


## Calculation 1/100
## Calculation specific variables:
##           sun_x = 1.0

### Calculation results:
###             F_c = [0.815607578905, 0.815607607469, 0.815607638611, 0.815607672567, ...
###            F_ib = [0.888035106304, 0.888035118346, 0.888035131474, 0.888035145789, ...
###             F_t = [0.813372049963, 0.811136549584, 0.808901051785, 0.806665556799, ...
###             F_v = [0.813372049963, 0.811136549584, 0.808901051785, 0.806665556799, ...
###            G_ic = [1.03672579047e+16, 1.03672612872e+16, 1.03672649752e+16, 1.0367 ...
###            G_vc = [3.90198936148e+21, 3.90198936148e+21, 3.90198936148e+21, 3.9019 ...
###            G_vi = [8.12258727337e+15, 8.12258614025e+15, 8.12258490478e+15, 8.1225 ...
###               J = [625.16886213, 625.16886213, 625.16886213, 625.16886213, 625.168 ...
###               P = [1.397583085, 2.79516617, 4.192749255, 5.59033234, 6.987915425,  ...
###            R_ic = [1.1883352228e+14, 1.18833636999e+14, 1.18833762079e+14, 1.18833 ...
###            R_vc = [5676041.13234, 6188719.10177, 6747703.76528, 7357177.6898, 8021 ...
###            R_vi = [2226099.16964, 2427165.59055, 2646392.60913, 2885420.48056, 314 ...
###          SRH_ic = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 ...
###          SRH_vc = [1.29572075406e-06, 2.95324117192e-06, 5.05132082539e-06, 7.6846 ...
###          SRH_vi = [0.00897331792717, 0.0107201841711, 0.0128016182702, 0.015281215 ...
###               V = [0.00223552894212, 0.00447105788423, 0.00670658682635, 0.0089421 ...
###          Vshift = [0.813372049963, 0.811136549584, 0.808901051785, 0.806665556799, ...
###        auger_ic = [2.12583711268e+15, 2.12584151397e+15, 2.12584631274e+15, 2.1258 ...
###      auger_ic_e = [2.12583710931e+15, 2.1258415103e+15, 2.12584630874e+15, 2.12585 ...
###      auger_ic_h = [3366460.82019, 3670529.68626, 4002062.91638, 4363541.16044, 475 ...
###        auger_vc = [2367.98591218, 2581.87273477, 2815.0790296, 3069.34989417, 3346 ...
###      auger_vc_e = [2367.98590843, 2581.87273031, 2815.0790243, 3069.34988787, 3346 ...
###      auger_vc_h = [3.74992596967e-06, 4.45792428873e-06, 5.29959448316e-06, 6.3001 ...
###        auger_vi = [1439393.55753, 1569404.6439, 1711158.8391, 1865716.83607, 20342 ...
###      auger_vi_e = [1439393.55525, 1569404.64119, 1711158.83587, 1865716.83224, 203 ...
###      auger_vi_h = [0.00227941359541, 0.00270977224658, 0.00322138307597, 0.0038295 ...
###          errors = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ...
###     ib_net_flux = [-59.5089733179, 33.9892798158, -9.76280161827, -70.7652812158,  ...
###               n = [2.46364717221e+14, 2.46364989422e+14, 2.46365286206e+14, 2.4636 ...
###            n_ib = [1.49754179257e+17, 1.49754228118e+17, 1.49754281391e+17, 1.4975 ...
###             n_t = [9.99994560257e+16, 9.99994068932e+16, 9.9999353323e+16, 9.99992 ...
###               p = [390141.447984, 425379.786287, 463800.873917, 505692.170094, 551 ...
###               η = [0.000875768303309, 0.00175153660662, 0.00262730490992, 0.003503 ...


## Calculation 2/100
## Calculation specific variables:
##           sun_x = 1.11455968399

### Calculation results:
...
...
```

Appendix C

Stray light plots

C.1 Stray light for 1000EX, 1000EM gratings

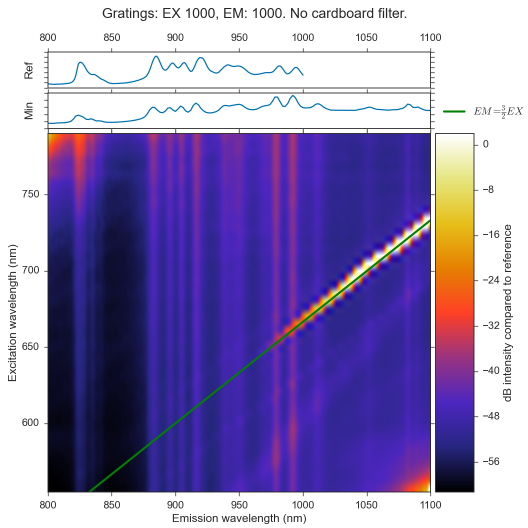

Figure C.1: Stray light measurement for the 1000 excitation grating and 1000 emission grating with no filters.

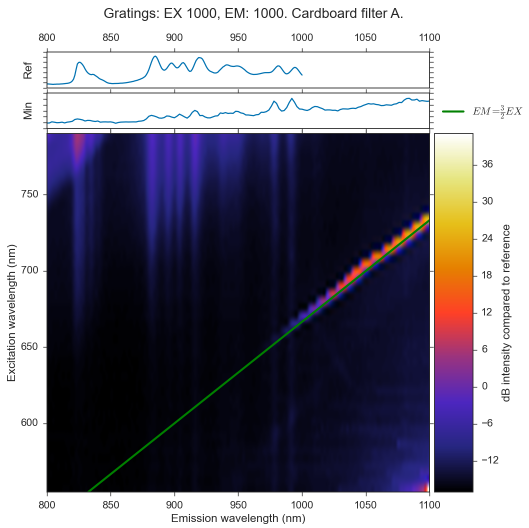

Figure C.2: Stray light measurement for the 1000 excitation grating and 1000 emission grating with filter A.

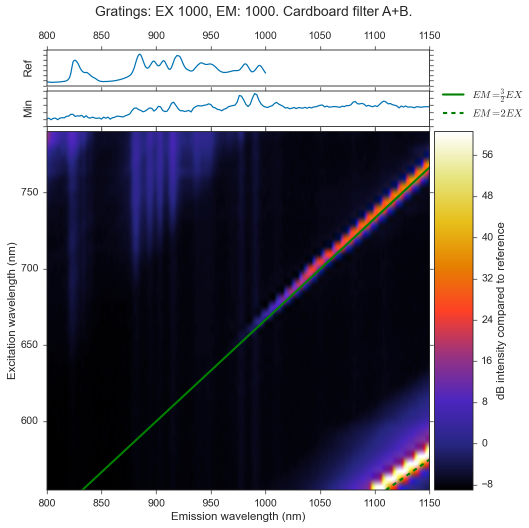

Figure C.3: Stray light measurement for the 1000 excitation grating and 1000 emission grating with filter A and B.

C.2 Stray light for C filter

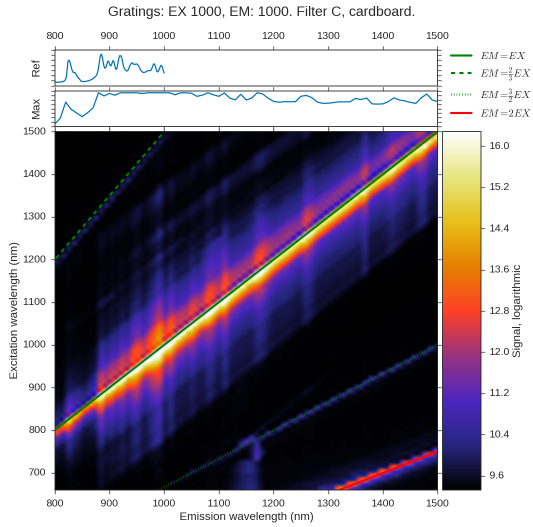

Figure C.4: Stray light measurement for the 1000 excitation grating and 1000 emission grating with filter C made of cardboard. It can be seen that there was some transmission through the cardboard.