MARMARA UNIVERSITY
FACULTY OF ENINEERING

THERMOPHOTOVOLTAIC (TPV) SYSTEM DESIGN AND ANALYSIS

FURKAN TURAN, AHMET BARIŞ DAĞISTANLI

GRADUATION PROJECT REPORT
Department of Mechanical Engineering

Supervisor
Assoc. Prof. Dr. Mustafa YILMAZ

ISTANBUL, 2020

MARMARA UNIVERSITY
FACULTY OF ENGINEERING

THERMOPHOTOVOLTAIC (TPV) SYSTEM DESIGN AND ANALYSIS

by

FURKAN TURAN, AHMET BARIŞ DAĞISTANLI

May 24, 2014, İstanbul

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

BACHELOR OF SCIENCE

AT

MARMARA UNIVERSITY

Signature of Author(s) ....
Department of Mechanical Engineering

Certified By .... Doç. Dr. Mustafa YILMAZ ....
Project Supervisor, Department of Mechanical Engineering

Accepted By..
Head of the Department of Mechanical Engineering

ACKNOWLEDGEMENT

First of all, I would like to thank my supervisor Assoc. Prof. Dr. Mustafa YILMAZ, for the valuable guidance and advice on preparing this thesis and giving me moral and material support.

July, 2020 Furkan TURAN, Ahmet Barış DAĞISTANLI

CONTENTS

ÖZET

Termofotovoltaik Sistem Optimizasyonu

Termofotovoltaikler son yıllarda yenilenebilir enerji sektörü için oldukça ilgi çekici bir mahiyet kazanmıştır. Bu rapor da termofotovoltaikler üzerine genel optimizasyon çalışmaları ihtiva etmektedir. Temel prensibi ısı enerjisini elektrik enerjisine dönüştürmek olan Termofotovoltaikler temel olarak emitör ve termofotovoltaik hücresi olmak üzere iki temel parçadan oluşmaktadır. Bu parçaların seçiminde emitörün yayım kabiliyeti, termofotovoltaik hücresinin uygun dalga boyu aralığı ve çalışma sıcaklıklarında olması temel kriterlerimizdir. Bunların yanı sıra termofotovoltaik hücresi ve emitör arasındaki mesafe, emitör alanı, termofotovoltaik hücresinin alanı ve soğutucu sistemin özellikleri sistemin güç çıkışını etkileyen temel kriterler arasındadır. Sistemin verimi ve güç yoğunluğu ise sıcaklık, spektrumların dalga boyu, foton akısı miktarı, iç yapılar arasındaki boşluklar ve sistemin tasarımı gibi etkenlere bağlıdır. Bu çalışma önceki cümlelerde belirtmiş olduğumuz malzemeleri ve etkenleri çeşitli konfigürasyonlar ile optimize edip optimum sistemi oluşturmayı amaçlamaktadır.

Bu çalışmada teorik hesaplamalar üzerinden termofotovoltaik panellerin MATLAB uygulamasında Fmincon, Big Bang and Big Crunch ve Particle Swarm optimizasyon yöntemlerini kullanarak optimizasyonlar yapacağız ve ilgili parametreleri belirledikten sonra bu parametrelerin güç çıktısı üzerindeki etkilerini gözlemleyeceğiz.

ABSTRACT

Thermophotovoltaic System Optimization

Thermophotovoltaics have gained a very interesting nature for the renewable energy sector in recent years. This report also includes general optimization studies on thermophotovoltaics. Thermophotovoltaics, whose basic principle is to convert heat energy into electrical energy, mainly consists of two basic parts, the emitter and the thermophotovoltaic cell. In the selection of these parts, the emission ability of the emitter, the appropriate wavelength range and operating temperatures of the thermophotovoltaic cell are our basic criteria. In addition, the distance between the thermophotovoltaic cell and the emitter, the emitter area, the area of the thermophotovoltaic cell, and the

characteristics of the cooling system are among the main criteria affecting the power output of the system. The efficiency and power density of the system depends on factors such as temperature, wavelength of the spectra, amount of photon flux, gaps between the internal structures and the design of the system. This study aims to optimize the materials and factors we mentioned in the previous sentences with various configurations and to create the optimum system.

In this study, we will perform optimizations by using Fmincon, Big Bang and Big Crunch and Particle Swarm optimization methods on MATLAB application of thermophotovoltaic panels over theoretical calculations and after determining the related parameters, we will observe the effects of these parameters on Output Work.

SYMBOLS

C_d : coefficient of derivative control

Ga : germanium or gallium

Sb : antimonide

P : power

A : surface area

ε : emissivity

σ : Stefan-Boltzmann constant

T : temperature

GaSb : gallium antimonide

Si : silicon

$T_{emitter}$: emitter temperature

T_{cell} : cell temperature

T_{water} : water temperature

$T_{coolant}$: coolant temperature

$A_{emitter}$: emitter surface area

A_{cell} : cell surface area

R_{conv} : thermal resistance for convection

R_{cyl} : thermal resistance for conduction

$R_{cyl, si}$: thermal resistance for conduction for si cell

$\mathbf{R_{cyl,\ GaSb}}$ **:** thermal resistance for conduction for GaSb cell

$\mathbf{R_{total}}$ **:** total thermal resistance

k **:** Boltzmann constant

$\mathbf{k_{water}}$ **:** thermal conductivity of water

$\mathbf{k_{si}}$ **:** thermal conductivity of si cell

$\mathbf{k_{GaSb}}$ **:** thermal conductivity of GaSb cell

$\mathbf{k_{pipe}}$ **:** thermal conductivity of pipe

$\mathbf{r_1}$ **:** inner radius of pipe

$\mathbf{r_2}$ **:** outer radius of pipe

Nu **:** nusselt number

Pr **:** prandtl number

$\mathbf{Pr_{water}}$ **:** prandtl number of water

ρ : density

µ : viscosity

V **:** velocity of water

Re **:** reynolds number

D **:** diameter of pipe

$\mathbf{L_1}$ **:** length of pipe

$\mathbf{L_2}$ **:** thickness of cell

$\mathbf{Q_{cool}}$ **:** power that transfers from cell to water

$\mathbf{I_{e\lambda}}$ **:** spectral intensity of radiation

$\mathbf{w}$: solid angle

$\boldsymbol{\lambda}$: wave length

$\mathbf{Q_{rad}}$: radiative power that transfer from emitter to cell

$\mathbf{m_{cell}}$: mass of cell

$\mathbf{c_{cell}}$: thermal conductivity of cell

$\mathbf{T_{c0}}$: initial temperature of cell

$\mathbf{n}$: cell efficiency

$\mathbf{c_0}$: speed of light in vacuum

$\mathbf{h}$: Planck constant

$\mathbf{Q_{output}}$: output power of thermophotovoltaic system

$\mathbf{Q_{output,\ exp}}$: experimental output power of thermophotovoltaic system

ABBREVIATIONS

TPV **:** Thermophotovoltaic

PV **:** Photovoltaic

RTPV **:** Radioisotope Thermophotovoltaic

STPV **:** Solar Thermophotovoltaic

PCG **:** Preconditioned Conjugated Gradient

PSO **:** Particle Swarm Optimization

A-life : Artificial Life

BB BC : Big Bang and Big Crunch

CHP : Combined Heat and Power

LIST OF FIGURES

PAGE

LIST OF TABLES

PAGE

1. INTRODUCTION

In photovoltaic systems, electricity is produced from photons occurring on the surface of the photovoltaic receptor cell. For photovoltaic systems, these photons are emitted by the sun, which can be considered as a heat source at a very large distance with a surface temperature of about 6000K. Similar to this process, it is possible to convert the light emitted by another heat source. Since the heat source is the primary energy source for this type of photovoltaic technology, the system is typically referred to as "thermophotovoltaic", which refers to the "thermo" heat source. The photovoltaic part then refers to the principle that the heat source radiates in "photons" where electricity or alternatively "voltaics" are produced by the receiving cells. Thermophotovoltaics are often abbreviated as TPV. TPV has a number of attractive features such as energy conversion, fuel versatility (nuclear, fossil, biomass, etc.), no noise, low maintenance, lightness, high power density and heat and power cogeneration as no moving parts are used. TPV can be used for potentially distributed power, combined heat and power (CHP), or for generating electricity from waste heat in the high temperature industries.

In addition to the heat source and receiver cells, extra elements are also used in general TPV systems to increase overall energy conversion efficiency. A selective emitter is used to transform the near blackbody radiation into a spectrum, rather than the TPV cell being directly illuminated by the heat source, which will cause undesirable heating of the cell due to the large amount of sub band gap photons. it is more suitable for the recipient cell.

When a selective emitter converts the incoming energy into a spectrum that better fits the absorbance of the recipient cells, many sub band gap photons will still reach the cell surface. Using a selective filter or a back-surface mirror or a combination, these sub band gap photons can be reflected back to the heat source.

Because the heat source is different from the sun, a different spectrum occurs on the cell surface than the solar photovoltaics. Compared to the temperature of the solar source, the lower temperature of the heat source (typically about 1200 ° C) will cause a longer shift in the emission spectrum. This shift requires the use of a low band semiconductor as the receiving cell to absorb low energy photons. Typical semiconductors used as absorbers in TPV systems are semiconductors based on compounds III-V containing germanium or gallium (Ga) and antimonide (Sb).

A comparison of TPV with PV, where an event energy density of 0.1Wcm - 2 (AM1.5G) is standard, is taken by a conventional TPV system at TPV energy densities in the range of 5-30Wcm - 2. This means that for a given electrical output power, a much smaller TPV system is needed, which makes the technology more practical and usable for mobile devices. Since sunlight is not used in TPV systems, this technology is not limited to the availability of sunlight only and electricity can be produced at the user's request.

Alternative heat sources are possible, as well as TPV systems based on a combustion heat source. An example is a TPV system in which electricity is generated from heat emitted by an isotope. This technology is called 'radioisotope thermophotovoltaics' (RTPV) and can be used for energy supply in deep space missions. For space tasks close to the sun, it is also possible to use high intensity, concentrated, sunlight to heat a selective emitter around which TPV cells are placed. This system is called solar thermophotovoltaic (STPV) and can be applied to the world as an alternative to terrestrial concentrated photovoltaic systems.

2. MATERIAL AND METHOD

2.1. Material

Thermophotovoltaic systems mainly consist of emitter, reflector and TPV cell parts. It is not mandatory to use reflectors from these parts. Reflector was not used in our study.

2.1.1. General system

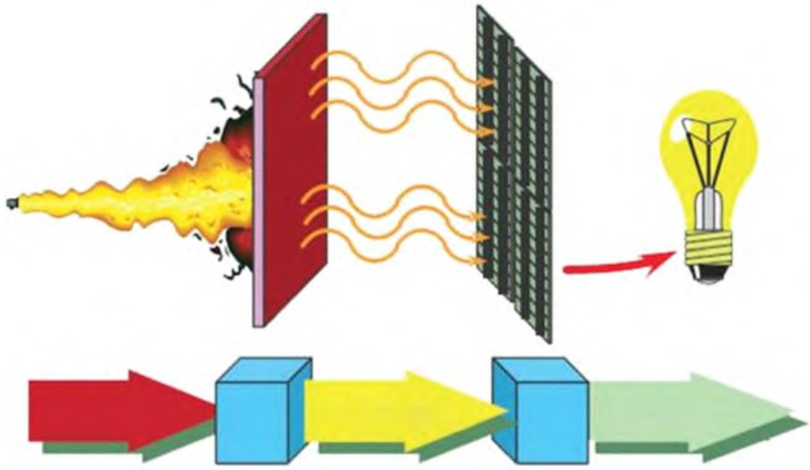

Figure 2.1 Schematic representation of thermophotovoltaic energy conversion.

The energy source used by the system is the heat source. This energy source can be any heat source, but generally waste heat is used as a heat source in factories to avoid extra cost. The emitter material radiates radially with the heat it receives from here. It converts the part of the radiation falling from the emitter to the TPV cell into TPV cell electrical energy. By adding reflector to the system, the irradiation of the wavelength that cannot be used can be returned to the emitter, thus increasing the efficiency of the system, but this process increases and complicates the production cost.

2.1.2. Heat source

There is no restriction on heat source in thermophotovoltaics systems. Heat sources generally consist of waste heat or heat sources already in nature. The reason for this is not to create an extra cost and to make low-efficiency thermophotovoltaic systems attractive. As can be seen from the power equation below, high heat emitting sources are preferred since the power obtained is proportional to the fourth power of the emitter temperature.

$$P = A \times \varepsilon \times \sigma \times T^4$$

Here A is the surface area (m^2), P is the total emitter power (W), σ is Stefan–Boltzmann constant and T is the temperature (K), ε the emissivity.

2.1.3. Emitter

Emitter is the part directly exposed to the heat source. The heat exposure causes the temperature of the emitter to increase. The emitter, whose temperature increases, makes radiative radiation in high amount and in various wavelengths. The radiation emitted from the emitter is tried to be dropped onto the TPV cell in the most efficient way. Emitter emits radiation at the rate of emissivity it has. Emitters with as high emissivity as possible are used in these systems. In this study, we assumed the emitter as a black body.
Work Output value is proportional to the square of the emitter emissivity value.

2.1.4. TPV cell

TPV converts the radiation from the cell emitter into electrical energy, but it can only convert radiation at certain wavelengths. This wavelength range varies depending on the TPV cell type. The efficiency of TPV cell is inversely proportional to the temperature of TPV cell. Usually a cooling system is used to lower the temperature of the TPV cell. In this study, we used Si and GaSb cells as TPV cells. In the cooling system, we used water as the refrigerant and polyurethane as the pipe.

2.2. Analyzed Cases

2.2.1. Case 1 and case 2

In the literature scans we made for TPV systems using GaSb Cell and Si Cell, the working conditions of the existing systems have been examined and the following limits have been determined in the optimizations to be made for our existing variables.

Table 2.1 Constraints of Parameters

Parameter	Minimum Value	Maximum Value
X_1 ($T_{emitter}$) (K)	800	2000
X_2 ($A_{emitter}$) (m)	0.0001	0.0001
X_3 (Distance) (m)	0.0001	0.02
X_4 (A_{cell}) (m^2)	0.0001	0.0001
X_5 (T_{water}) (K)	278	350
X_6 (T_{cell}) (K)	298	500

The optimization methods mentioned below will be used to find the maximum Work Output value according to the parameters mentioned above and the results will be compared with each other.

- Fmincon Optimization
- Particle Swarm Optimization
- Big Bang and Big Crunch Optimization

Emitter is accepted as black body and related optimization methods will be applied separately for Si Cell and GaSb Cell TPV systems.

2.2.2. Case 3 and case 4

The optimization methods we used in Case 1 and Case 2 will be compared with the experimental studies whose data are given below, and their consistency will be observed.

Table 2.2 Experimental values of parameters

	GaSb Cell	Si Cell
X_1 ($T_{emitter}$) (K)	1180	1230.8
X_2 ($A_{emitter}$) (m^2)	0.00073	0.001
X_3 (Distance) (m)	0.011	0.011
X_4 (A_{cell}) (m^2)	0.000146	0.0002
X_6 (T_{cell}) (K)	343.4	367.6

Emissivity values will be assumed to be 0.7.

2.3 METHOD

2.3.1 Formulization

TPV cell is heated by the emitter's radiation. We try to minimize this warming with a cooling system. We calculate the final temperature of the TPV cell by writing the necessary heat transfer equations. When making heat transfer calculations, we accept our TPV system under vacuum. The TPV cell temperature we found affects the TPV cell efficiency inversely proportional. While calculating the power output of our system, we consider how much of the radiation emitted by the emitter reaches our TPV cell and how efficiently this radiation is used by our TPV cell.

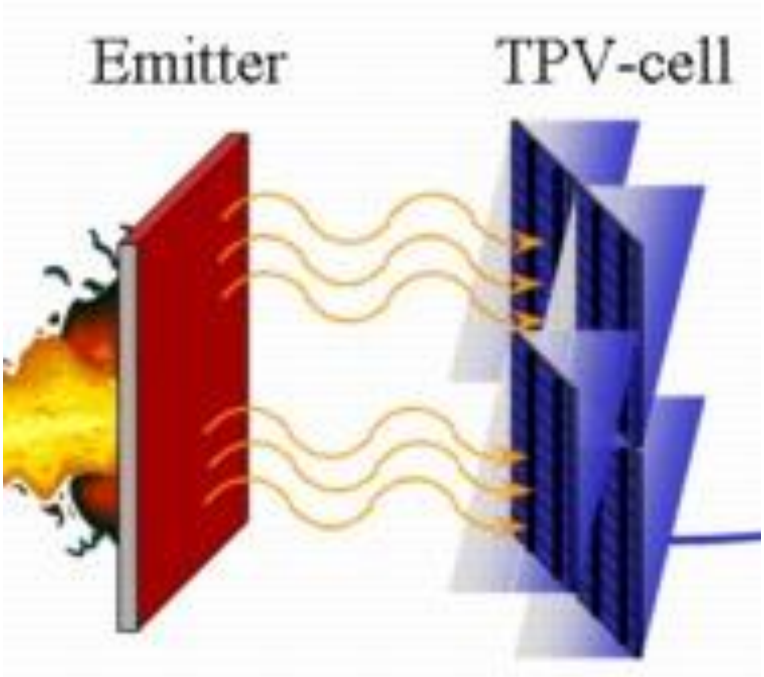

Figure 2.2 Radiation heat transfer example

Since the efficiency of the photovoltaic cells in our system decreases as the temperature increases, we use the cooling mechanism to lower the temperature of the photovoltaic cells. We used water as the coolant. We found it appropriate to use polyurethane material in the pipes where the coolant passes. We have seen that there are 3 types of heat transfer in our system since there is a flow inside the pipes and the pipes are adjacent to PV cells in the form of walls.

- Heat transfer in the part where refrigerant passes: Forced Convection
- Heat transfer at the wall thickness of the pipe: Conduction
- Heat transfer at the part where the PV cell and pipe contact: Conduction

In systems containing different heat transfers; 'Thermal Resistance' method is used to find the total heat transfer account in the system. According to the existing heat transfer type, that system has thermal resistance and while this method is applying separately for each heat transfer type and thermal resistance of the system is found. According to heat transfer types (convection, conduction, radiation), the formulas used to find thermal resistance vary and are as follows:

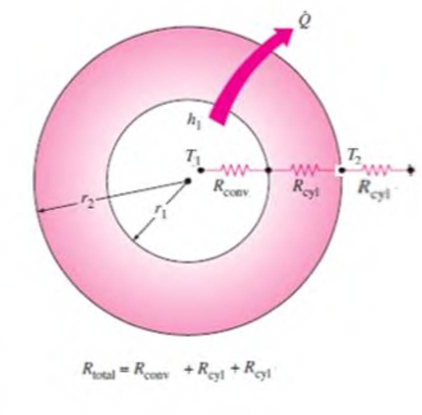

Figure 2.3a Thermal resistance method schematic view

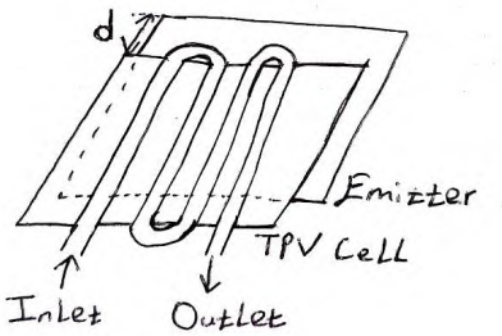

Figure 2.3b Integration of Cooling System

Thermal Resistance Formula at Forced Convection:

$$R_{conv} = \frac{1}{(2\pi r_1\ L_1)h_1}$$

Heat Coefficient of Forced Convection (h1) Formula:

$$h_1 = \frac{Nu * k_{water}}{D}$$

Nusselt Number Formula:

$$Nu = 0.332 * Pr^{\left(\frac{1}{3}\right)} * Re^{\left(\frac{1}{2}\right)}$$

Reynolds Number Formula:

$$Re = \frac{\rho * V * D}{\mu}$$

Thermal Resistance Formula at Conduction at the wall thickness of the pipe:

$$R_{cyl} = \frac{\ln\left(\frac{r_1}{r_2}\right)}{(2\pi L_1)k_{pipe}}$$

Thermal Resistance Formula at Conduction at the part where the PV cell and pipe contact:

$$R_{cyl,si} = \frac{L_2}{A_{cell} * k_{si}}$$

$$R_{cyl,GaSb} = \frac{L_2}{A_{cell} * k_{GaSb}}$$

The formula used to find R_{total}:

$$R_{total} = R_{conv,1} + R_{cyl} + R_{cyl}$$

The Formula used to find Q_{cool}:

$$Q_{cool} = \frac{T_{cell} - T_{water}}{R_{total}}$$

The spectral intensity of radiation emitted by an emitter at an absolute temperature T at a wavelength λ has been determined by Max Planck;

$$I_{e\lambda}(\lambda, T) = \frac{2hc_0{}^2}{\lambda^5\left[exp\left(\frac{hc_0}{\lambda kT}\right) - 1\right]} \quad \left(\frac{W}{m^2 * sr * \mu m}\right)$$

The solid angle between emitter and TPV cell;

$$w = \frac{A_{cell}}{d^2} \quad (sr)$$

The heat transfer from emitter to TPV cell by radiation;

$$Q_{rad} = \varepsilon\sigma A_{emitter}\left(T_{emitter}{}^4 - T_{cell}{}^4\right) (W)$$

The heat transfer from TPV cell to cooling system;

$$Q_{cool} = \frac{T_{cell} - T_{water}}{R_{total}} \quad (W)$$

Net heat transfer for TPV cell;

$$Q_{rad} - Q_{cool} = m_{cell}c_{cell}(T_{cell} - T_{c0}) (W)$$

Final TPV cell temperature;

$$T_{cell} = \frac{\varepsilon\sigma A_{emitter}\left(T_{emitter}{}^4 - T_{cell}{}^4\right) + \frac{T_{water}}{R_{total}} + mcT_{c0}}{\frac{1}{R_{total}} + m_{cell}c_{cell}} \quad (K)$$

The efficiency of TPV cell at 1300K;

$$n = -0.0021 \frac{\varepsilon\sigma A_{emitter}\left(T_{emitter}{}^4 - T_{cell}{}^4\right) + \frac{T_{water}}{R_{total}} + m_{cell}c_{cell}T_{c0}}{\frac{1}{R_{total}} + m_{cell}c_{cell}} + 0.848$$

Output work of the TPV system;

$$Q_{output} = \frac{2hc_0{}^2}{\lambda^5\left[exp\left(\frac{hc_0}{\lambda kT}\right) - 1\right]} * \frac{A_{cell}}{d^2} * A_{emitter} * n * \varepsilon \; (W)$$

2.4. Optimization

2.4.1. Fmincon optimization method

Starting with the first guess, fmincon finds a restricted minimum scalar function of various variables. This is often called constrained nonlinear optimization or nonlinear programming.

x = fmincon (fun, x0, A, b) starts at x0 and finds a minimum x, subject to the function defined in fun, A * x <= b linear inequalities. x0 can be a scalar, vector or matrix.

x = fmincon (fun, x0, A, b, Aeq, beq), Aeq * x = beq and A * x <= b minimizes entertainment subject to linear equations. If there is inequality, set A = [] and b = [].

x = fmincon (fun, x0, A, b, Aeq, beq, lb, ub) defines a set of lower and upper bounds on the design variables x, so the solution is always in the lb range <= x <= ub. If there is no equation, set Aeq = [] and beq = [].

x = fmincon (fun, x0, A, b, Aeq, beq, lb, ub, nonlcon) minimizes nonlinear inequalities defined in nonlcon to c (x) or equality ceq (x). fmincon optimizes to be c (x) <= 0 and ceq (x) = 0. If there is no limit, set lb = [] and / or ub = [].

x = fmincon (fun, x0, A, b, Aeq, beq, lb, ub, nonlcon, options) is minimized with the optimization parameters specified in the build options. Use optimset to set these parameters.

x = fmincon (fun, x0, A, b, Aeq, beq, lb, ub, nonlcon, options, P1, P2, ...) problem-dependent parameters P1, P2, etc. passes directly to fun and nonlcon. If these arguments are not needed, pass blank matrices as A, b, Aeq, beq, lb, ub, nonlcon, and placeholders for options.

[x, fval] = fmincon (...) returns the value of objective function fun in x solution.

[x, fval, exitflag] = fmincon (...) returns a value that describes the exit condition of fmincon, exitflag. [x, fval, exitflag, output] = fmincon (...) returns a struct output containing information about optimization.

[x, fval, exitflag, output, lambda] = fmincon (...) returns a structure lambda that contains the Lagrange multipliers in the x solution.

[x, fval, exitflag, output, lambda, grad] = fmincon (...) returns the value of the entertainment slope in the x solution.

[x, fval, exitflag, output, lambda, grad, hemp] = fmincon (...) returns the value of Hessian's fun solution, x.

By default, fmincon selects the large-scale algorithm if the user provides gradient in Fun and only has upper and lower limits, or only linear equality constraints. This algorithm is a subspace confidence zone method and is based on the internal reflective Newton method. Each iteration contains an approximate solution of a large linear system using the preconditioned conjugated gradients (PCG) method.

Fmincon optimization aims to find the minimum value of the function. Since we want to achieve maximum Work Output, we multiply our function by -1 and put it into fmincon optimization. We get the Work Output value by taking the absolute value of the fval value.

2.4.1.1. Codes of Fmincon optimization

Editor - C:\Users\furkan\Desktop\Bitirme_Opt\FMINCON\verimopt.m

fminverim.m | fminverimdd.m | verimopt.m | +

```
function [f,df] = verimopt(x)
T=298;
m=0.244*10^(-3);;%% density of Si = 2.3290 g/cm^3  , density of GaSb= 5.61 g/cm^3 ,
%% m of Si = 0.09316*(10^-3), m of GaSb = 0.244*(10^-3)
s=250;%% Ssi=710 SGaSb= 250
R=2.82;%% Rsi =2.72, RGaSb= 2.82, kGaSb= 32 , k Si = 150
f=-((1.191*10^8/(1.7^5*(exp(1.439*10^4/1.7/x(1))-1))))*...
    (-0.0021*(5.6703*10^(-8)*x(4)*(x(1)^4-x(6)^4)+x(5)/R+m*s*T)/((1/R)+m*s)+0.848)*x(2)/x(3)^2*x(4);
df(1)= (0.00399533*x(1)^3*x(2)*x(4)^2)/...
    ((exp(8464.71/x(1))-1)*x(3)^2*(m*s+1/x(4)))-(7.10034*10^10*exp(8464.71/x(1))*x(2)...
    *x(4)*(0.848-(0.0021*(5.6703*10^(-8)*x(4)*(x(1)^4-x(6)^4)+m*s*x(5)+x(5)/x(4)))/...
    (m*s+1/x(4))))/(x(1)^2*(exp(8464.71/x(1)) - 1)^2*x(3)^2);
df(2)= -(8.38817*10^6*x(4)*(0.848-(0.0021*(5.6703*10^(-8)*x(4)*(x(1)^4-x(6)^4)+m*s*x(5)+x(5)/x(4)))/...
    (m*s+1/x(4))))/((exp(8464.71/x(1))-1)*x(3)^2);
df(3)= (1.67763*10^7*x(2)*x(4)*(0.848-(0.0021*(5.6703*10^(-8)*x(4)*...
    (x(1)^4-x(6)^4)+m*s*x(5)+x(5)/x(4)))/(m*s+1/x(4))))/((exp(8464.71/x(1))-1)*x(3)^3);
df(4)= (x(2)*(0.00199766*x(1)^4*x(4)*x(4)-0.00199766*x(6)^4*x(4)*x(4)+17615.2*m*x(4)...
    *s*x(5)-7.11317*10^6*m*x(4)*s+17615.2*x(5)-7.11317*10^6))/((exp(8464.71/x(1))-1)*x(3)^2*...
    (m*x(4)*s+1));
df(5)= (17615.2*x(2)*x(4))/((-1+exp(8464.71/x(1)))*x(3)^2*(1+s*m*x(4)));
df(6)= -(0.00399533*x(2)*x(6)^3*x(4)^2*x(4))/((-1+exp(8464.71/x(1)))*x(3)^2*(1+s*m*x(4)));
end
```

Figure 2.4 verimopt.m matlab file codes

Editor - C:\Users\furkan\Desktop\Bitirme_Opt\FMINCON\fminverimdd.m

fminverim.m | fminverimdd.m | verimopt.m | +

```
function [c,ceq] = fminverimdd(x)
%   x(1)=T_emitter, x(2)=A_cell, x(3)=Distance, x(4)=A_emitter, x(5)=T_coolant, x(6)=T_cell
T=298;
m=0.09316*10^(-3);%% density of Si = 2.3290 g/cm^3  , density of GaSb= 5.61 g/cm^3 ,
%% m of Si = 0.09316*(10^-3), m of GaSb = 0.244*(10^-3)
s=710;%% Ssi=710 SGaSb= 250
R=2.72;%% Rsi =2.72, RGaSb= 2.82, kGaSb= 32 , k Si = 150
c = (5.6703*10^(-8)*x(4)*(x(1)^4-x(6)^4)+x(5)/R+m*s*T)/((1/R)+m*s)-x(6);
ceq = [];
end
```

Figure 2.5 fminverimdd.m matlab file codes

```
%   x(1)=T_emitter, x(2)=A_cell, x(3)=Distance, x(4)=A_emitter, x(5)=T_coolant, x(6)=T_cell
A = [];
b = [];
Aeq = [];
beq = [];
lb = [1230.8,0.001,0.011,0.0002,300,367.6];
ub = [1230.8,0.001,0.011,0.0002,350,367.6];
x0 = (lb + ub)/2;
fun = @Si;
nonlcon = @fminverimdd;
Q_out = fval;
[x,Q_out] = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)
```

Figure 2.6 fminverim.m matlab file codes

2.4.2. Particle swarm optimization method

Particle swarm optimization has its roots in two main component methodologies. Perhaps, more generally, their ties to artificial life (A-life) and especially to bird flock, fish school, and herd theory are more obvious. However, it is also associated with evolutionary computation and has links to both genetic algorithms and evolutionary programming. [3] This algorithm explores the space of an objective function by setting the trajectories of individual agents, called particles, as segmented paths created by position vectors in a semi-stochastic way.

The motion of a lot of particles consists of two main components: a stochastic component and a deterministic component. While each particle is drawn towards the current global best g^* and its best position in history x_i^*, there is also a tendency to move randomly. Let x and vi be the position vector and velocity for particle i, respectively. The new velocity vector is determined by the formula

$$v_i{}^{t+1} = v_i{}^{t} + \alpha * \varepsilon 1 * \times \ (g \, * \, - x_i^t) + \ \beta * \ \varepsilon_2 * (x_i^* - x_i^t)$$

where ε1 and ε2 are two random vectors, and each input takes values between 0 and 1. α and β parameters, typically learning parameters or acceleration constants that can be taken as

a ≈ β ≈ 2.

The starting locations of all particles should be relatively uniformly distributed so that they can take samples from many regions, which is especially important for multimodal problems. The initial velocity of a particle can be taken as zero, i.e. $v_i^t = 0$. New then updated with

$$x_i^{t+1} = x_i^t + v_i{}^{t+1}$$

There are many variants which extend the standard PSO algorithm (Kennedy et al., 2001; Yang, 2008; Yang, 2010b), and the most noticeable improvement is probably to use inertia function θ(t) so that, ${v_i}^t$ is replaced by θ(t) x ${v_i}^t$

$$ {v_i}^{t+1} = \theta(t) * {v_i}^t + \alpha * \varepsilon 1 * \times \ (g \ * \ -x_i^t) + \ \beta * \ \varepsilon_2 * (x_i^* - x_i^t) $$

where θ takes the values between 0 and 1. In the simplest case, the inertia function can be taken as a constant, typically θ ≈ 0.5 ∼ 0.9. This is equivalent to introducing a virtual mass to stabilize the motion of the particles, and thus the algorithm is expected to converge more quickly.[4]

2.4.2.1. Codes of particle swarm optimization

pso2_optimization.m | PSO.m | Optimization.m

```
clc;
clear;
close all;

%% Problem Definiton

problem.CostFunction = @(x) Optimization(x);  % Cost Function
problem.nVar = 6;       % Number of Unknown (Decision) Variables
problem.VarMin = [1230.8 0.001 0.011 0.0002 343 367.6];   % Lower Bound of Decision Variables
problem.VarMax = [1230.8 0.001 0.011 0.0002 400 367.6];    % Upper Bound of Decision Variables

%% Parameters of PSO

% Constriction Coefficients
kappa = 1;
phi1 = 2.05;
phi2 = 2.05;
phi = phi1 + phi2;
chi = 2*kappa/abs(2-phi-sqrt(phi^2-4*phi));

params.MaxIt = 100;        % Maximum Number of Iterations
params.nPop = 50;           % Population Size (Swarm Size)
params.w = chi;             % Intertia Coefficient
params.wdamp = 1;           % Damping Ratio of Inertia Coefficient
params.c1 = chi*phi1;       % Personal Acceleration Coefficient
params.c2 = chi*phi2;       % Social Acceleration Coefficient
params.ShowIterInfo = true; % Flag for Showing Iteration Informatin

%% Calling PSO

out = PSO(problem, params);

BestSol = out.BestSol;
BestCosts = out.BestCosts;

%% Results

figure;
semilogy(BestCosts, 'LineWidth', 2);
xlabel('Iteration');
ylabel('Best Cost');
grid on;
```

Figure 2.7 pso2_optimization.m matlab file codes

PSO.m pso2_optimization.m

```
function out = PSO(problem, params)

    %% Problem Definiton

    CostFunction = problem.CostFunction;  % Cost Function

    nVar = problem.nVar;          % Number of Unknown (Decision) Variables

    VarSize = [1 nVar];           % Matrix Size of Decision Variables

    VarMin = problem.VarMin;      % Lower Bound of Decision Variables
    VarMax = problem.VarMax;      % Upper Bound of Decision Variables


    %% Parameters of PSO

    MaxIt = params.MaxIt;     % Maximum Number of Iterations

    nPop = params.nPop;       % Population Size (Swarm Size)

    w = params.w;             % Intertia Coefficient
    wdamp = params.wdamp;     % Damping Ratio of Inertia Coefficient
    c1 = params.c1;           % Personal Acceleration Coefficient
    c2 = params.c2;           % Social Acceleration Coefficient

    % The Flag for Showing Iteration Information
    ShowIterInfo = params.ShowIterInfo;

    MaxVelocity = (VarMax-VarMin);
    MinVelocity = -MaxVelocity;

    %% Initialization

    % The Particle Template
    empty_particle.Position = [];
    empty_particle.Velocity = [];
    empty_particle.Cost = [];
    empty_particle.Best.Position = [];
    empty_particle.Best.Cost = [];

    % Create Population Array
    particle = repmat(empty_particle, nPop, 1);

    % Initialize Global Best
    GlobalBest.Cost = -inf;

```

Figure 2.8 pso.m matlab file codes row 1-46

```
    % Initialize Population Members
    for i=1:nPop

        % Generate Random Solution
        particle(i).Position = unifrnd(VarMin, VarMax, VarSize);

        % Initialize Velocity
        particle(i).Velocity = zeros(VarSize);

        % Evaluation
        particle(i).Cost = CostFunction(particle(i).Position);

        % Update the Personal Best
        particle(i).Best.Position = particle(i).Position;
        particle(i).Best.Cost = particle(i).Cost;

        % Update Global Best
        if particle(i).Best.Cost > GlobalBest.Cost
            GlobalBest = particle(i).Best;
        end


    end

    % Array to Hold Best Cost Value on Each Iteration
    BestCosts = zeros(MaxIt, 1);


%% Main Loop of PSO

for it=1:MaxIt

        for i=1:nPop

        % Update Velocity
        particle(i).Velocity = w*particle(i).Velocity ...
            + c1*rand(VarSize).*(particle(i).Best.Position - particle(i).Position) ...
            + c2*rand(VarSize).*(GlobalBest.Position - particle(i).Position);

        % Apply Velocity Limits
        particle(i).Velocity = max(particle(i).Velocity, MinVelocity);
        particle(i).Velocity = min(particle(i).Velocity, MaxVelocity);

        % Update Position
        particle(i).Position = particle(i).Position + particle(i).Velocity;

```

Figure 2.9 pso.m matlab file codes row 47-92

```
                % Apply Lower and Upper Bound Limits
                particle(i).Position = max(particle(i).Position, VarMin);
                particle(i).Position = min(particle(i).Position, VarMax);

                % Evaluation
                particle(i).Cost = CostFunction(particle(i).Position);

                % Update Personal Best
                     if particle(i).Cost > particle(i).Best.Cost

                    particle(i).Best.Position = particle(i).Position;
                    particle(i).Best.Cost = particle(i).Cost;

                    % Update Global Best
                        if particle(i).Best.Cost > GlobalBest.Cost
                                 GlobalBest = particle(i).Best;
                        end

                    end
                end

        % Store the Best Cost Value
        BestCosts(it) = GlobalBest.Cost;

        % Display Iteration Information
        if ShowIterInfo
            disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCosts(it))]);
        end

        % Damping Inertia Coefficient
        w = w * wdamp;

    end

    out.pop = particle;
    out.BestSol = GlobalBest;
    out.BestCosts = BestCosts;
end
```

Figure 2.10 pso.m matlab file codes row 93-130

pso2_optimization.m | PSO.m | Optimization.m | +

```
function z = Optimization(x)

    x1=x(1,1);
    x2=x(1,2);
    x3=x(1,3);
    x4=x(1,4);
    x5=x(1,5);
    x6=x(1,6);
    T=298;
    m=0.09316*10^(-3); %% density of Si = 2.3290 g/cm^3  , density of GaSb= 5.61 g/cm^3 ,
%% m of Si = 0.09316*(10^-3), m of GaSb = 0.244*(10^-3)

    s=710; %% Ssi=710 SGaSb= 250
    R=2.72; %% Rsi =2.72, RGaSb= 2.82, kGaSb= 32 , k Si = 150
    of = ((1.191*10^8/(1.7^5*(exp(1.439*10^4/1.7/x1)-1))))*...
        (-0.0021*(5.670374419*10^(-8)*x4*(x1^4-x6^4)+x5/R+m*s*T)/((1/R)+m*s)+0.848)*x2/x3^2*x4;

```

Figure 2.11 Optimization.m matlab file codes row 1-17

```
18 -        constraints = [];
19
20 -        constraints(1,1) = (5.670374419*10^(-8)*x4*(x1^4-x6^4)+x5/R+m*s*T)/((1/R)+m*s)-x6;
21 -        p = 1:length(constraints);
22
23 -        for i=1:length(constraints)
24
25 -            if constraints(i)<=40 && constraints(i)>=-40
26 -               p(i)=0;
27 -            else
28 -               p(i)=1;
29 -            end
30
31 -        end
32
33 -        penalty = -10000;
34
35 -        z = of+penalty*sum(p);
36 -    end
```

Figure 2.12 optimization.m matlab file codes row 18-36

2.4.3. Big bang and big crunch optimization

The Big Bang – Big Crunch algorithm is an algorithm for multivariable non-linear and linear problem optimization. It does not require any gradient operations and it is a solely a numeric approach.

The algorithm takes its name from the BB-BC idea of the universe where universe constantly expanding with big bang and dissipating energy, then crunching with gravity. The method firstly creates a random cluster of vectors within the given boundaries of the search space. This process id equivalent with the Big Bang phase.

After this initiation, fitting values of the initial candidate vector cluster are calculated. Then, with assumption of problem type is minimization, the mass center is calculated such that smaller mass center. This process is equivalent to the Big Crunch phase. Finally, the new cluster is created about the mass center and Big Bang and the Big Crunch are follow each other in a loop until a desired relative error or a maximum of iteration number is achieved. Alternatively, of the mass center the best with minimum value can be selected as new start point too.

However, in this piece of MATLAB code it is decided to use a hybrid way such that the new cluster is crated about weighted some of the best fit and the mass center.

It is observed that the best fit approach has more freedom about searching space whereas mass center approach is more stable and can focus too quickly to candidate position.

The formula used to calculate the mass center is as follows: [2] $xc=\Sigma 1 fixiNi=1 \Sigma 1 fiNi=1$

In this equation x stands for candidate vectors, N stands for the cluster population, xc stands for centroid vector and f stands for fitting values.

2.4.3.1. Codes of big bang and big crunch optimization

BB_BC_Method.m | f_bcorrector.m | f_bfit.m | f_mcenter.m | +

```
clc;clear;clf;
% --ATTENTION !-- Used functions are defined in seperate .m files
% Problem functions are defined in the following section
%% Problem Choice Section
    %%
    % Define objective function
    T=298;
    m=0.09316*10^(-3);
    s=250;
    R=2.72;
    f_e04=@(x) -((1.191*10^8/(1.7^5*(exp(1.439*10^4/1.7/x(1))-1))))*...
        (-0.0021*(5.670374419*10^(-8)*x(4)*(x(1)^4-x(6)^4)+x(5)/R+m*s*T)...
        /((1/R)+m*s)+0.848)*x(2)/x(3)^2*x(4);
    % Define constraints
    g1=@(x) (5.670374419*10^(-8)*x(4)*(x(1)^4-x(6)^4)+x(5)/R+m*s*T)/((1/R)+m*s)-x(6);
%     g2=@(x) (4*x(2)^2-x(1)*x(2))/(12566*x(2)*x(1)^3-x(1)^4)+1/(5108*x(1)^2)-1;
%     g3=@(x) 1-140.45*x(1)/(x(2)^2*x(3));
%     g4=@(x) (x(1)+x(2))/1.5-1;
    % Approximate objective function with penalty functions
    alfa=2;
    f=@(x,alfa) f_e04(x) +alfa*min(0,g1(x))^2;
    % Boundry matrix definitions
    bon_mat=[1000 2000;0.0001 0.0001;0.001 0.02;0.0001 0.0001;278 300;298 500];
    [n,~]=size(bon_mat);
    N=30; % Cluster size
    disp('Invalid problem number or invalid entry !')
%==========================================================================

%% Initial Cluster and best fit
%  Creating initial random cluster with respecting to boundries
  mu_vec=mean(transpose(bon_mat));
  sigma_vec=(bon_mat(:,2)-bon_mat(:,1))/2;
  lower_b=bon_mat(:,1);
  upper_b=bon_mat(:,2);
  rand_nums=zeros(N,n);
  for i=1:n
  rand_nums(:,i)=normrnd(mu_vec(i),0.75*sigma_vec(i),[N,1]); %
  end

  rand_nums=f_bcorrector(rand_nums,n,bon_mat); % Calling boundry corrector

  best_fit=f_bfit(rand_nums,N,f,alfa); % Calling f_bfit for finding best fit

  mass_center=f_mcenter(rand_nums,N,n,f); % Calling f_mcenter for mass center calc


    % define beta as 0.5
    beta=0.6;
     scatter(rand_nums(:,1),rand_nums(:,2),'x'); % plotting of initial cluster
     hold on; plot(best_fit(1),best_fit(2),'*','Color','Black','MarkerSize',8);
     xlabel('X1'); ylabel('X2'); pause(0.01);
```

Figure 2.13 BB_BC_Method.m matlab file codes row 1-51

```
%% ----------------------------MAIN LOOP-----------------------------------------------
for k=1:1:1000
   mu_vec=mean(rand_nums);
   sigma_vec=(upper_b-lower_b)/2/k;
     alfa=k*200;
   for i=1:n
       rand_nums(:,i)= (beta*mass_center(i)+(1-beta)*best_fit(i)) ...
           + 3*normrnd(0,sigma_vec(i),[N,1]);
   end

     rand_nums=f_bcorrector(rand_nums,n,bon_mat); % Calling boundry corrector

     best_fit=f_bfit(rand_nums,N,f,alfa); % Calling f_bfit for finding best fit

     mass_center=f_mcenter(rand_nums,N,n,f); % Calling f_mcenter for mass center calc

      fprintf('Iteration: %d , point:',k); disp(best_fit); disp(f(best_fit,alfa))

%        scatter(rand_nums(:,1),rand_nums(:,2),'x');
%        hold on; plot(best_fit(1),best_fit(2),'*','Color','Black','MarkerSize',8);
%        xlabel('X1'); ylabel('X2'); pause(0.01);

end
```

Figure 2.14 BB_BC_Method.m matlab file codes row 52-74

```
function rand_nums=f_bcorrector(rand_nums,n,bon_mat)
    for i=1:n
        col_ith=rand_nums(:,i);
        col_ith(col_ith < bon_mat(i,1))= bon_mat(i,1); % Min Boundry Correction
        col_ith(col_ith > bon_mat(i,2))= bon_mat(i,2); % Max Boundry Correction
        rand_nums(:,i)=col_ith;
    end
end
```

Figure 2.15 f_bcorrector.m matlab file codes

```

function best_fit=f_bfit(rand_nums,N,f,alfa)
    fits=zeros(N,1);
    for i=1:N
        fits(i)=f(rand_nums(i,:),alfa);
    end
    bfit_place=fits==min(fits);
    best_fit=rand_nums(bfit_place,:);

end
```

Figure 2.16 f_bit.m matlab file codes

```
function mass_center=f_mcenter(rand_nums,N,n,f)
    cent_vec_num=zeros(N,n); cent_vec_den=zeros(N,n);
    for i=1:N  % Keep calculating centroid
        cent_vec_num(i,:)=rand_nums(i,:)/f(rand_nums(i,:),20);
        cent_vec_den(i,:)=1./f(rand_nums(i,:),20);
    end
     mass_center=sum(cent_vec_num)./sum(cent_vec_den);
end
```

Figure 2.17 f_mcenter.m matlab file codes

3. RESULTS AND DISCUSSION

3.1. Heat Transfer Calculation's Results:

In the equations we gave in the Method section, we obtained the results by placing the following values in their places. We chose polyurethane as the pipe material from which the coolant flows because it is cold resistant and also widely used. We designated the refrigerant as water at 5 degrees. We determined Pr_{water}, μ, ρ and k_{water} values according to the properties of water at 5 degrees.

Table 3.1 Cooling system heat transfer parameters

k_{pipe} = 0.024 (W/mK)
k_{si} = 150 (W/mK)
k_{GaSb} = 32(W/mK)
k_{water} = 0.4975(W/mK)
Pr_{water} = 5
V = 3 (m/s)
ρ = 999.93 (kg/m^3)
μ = 0015125 (N/sm^2)
D = 0.01 (m)
L_1= 1 (m)
L_2= 1 (m)

$$Re = \frac{999.93 * 3 * 0.01}{0.0015125} = 19760$$

$$Nu = 0.332 * 5^{\left(\frac{1}{3}\right)} * 19760^{\left(\frac{1}{2}\right)} = 107$$

$$h_1 = \frac{107 * 0.4975}{0.01} = 5328 \left(\frac{W}{m^2 K}\right)$$

$$R_{conv,1} = \frac{1}{2\pi\left(\frac{0.01}{2}\right)(1)5328} = 5.97 * 10^{-3} \ (°\mathrm{C/W})$$

$$R_{cyl} = \frac{\ln\left(\frac{0.15}{0.10}\right)}{(2\pi 1)(0.024)} = 2.69 \ (°\mathrm{C/W})$$

$$R_{cyl,si} = \frac{0.4 * 10^{-3}}{(0.01)^2 * 150} = 0.027 \ (°\mathrm{C/W})$$

$$R_{cyl,GaSb} = \frac{0.4 * 10^{-3}}{(0.01)^2 * 32} = 0.125 \ (°\mathrm{C/W})$$

$$R_{total} = 5.97 * 10^{-3} + 2.69 + 0.125 = 2.82 \ for\ GaSb\ Cell$$

$$R_{total} = 5.97 * 10^{-3} + 2.69 + 0.027 = 2.72 \ for\ Si\ Cell$$

3.2. Case – 1: General Optimization of GaSb Cell

We kept the limits of the parameters in wide ranges to make a general optimization. We obtained the results from the 3 optimization methods we used. We compared these results we obtained. By taking A_{cell} and $A_{emitter}$ values 1 cm^2, we get the power output value per cm^2. These processes are calculated according to the features of GaSb cell.

Table 3.2 Optimization constraints and results for GaSb cell TPV system

	Fmincon Opt.	PSO Opt.	BB BC Opt.
$800<X_1$ ($T_{emitter}$) <2000 (K)	1517.0	1515.0	1516.0
$0.0001<X_2$ ($A_{emitter}$) <0.0001 (m^2)	0.0001	0.0001	0.0001
$0.001<X_3$ (Distance) <0.02 (m)	0.0010	0.0010	0.0010
$0.0001<X_4$ (A_{cell}) <0.0001 (m^2)	0.0001	0.0001	0.0001
$278<X_5$ (T_{water}) <350 (K)	278.00	278.00	278.00
$298<X_6$ (T_{cell}) <500 (K)	500.00	354.00	352.59
Q_{output} (W)	34.339	33.914	33.912

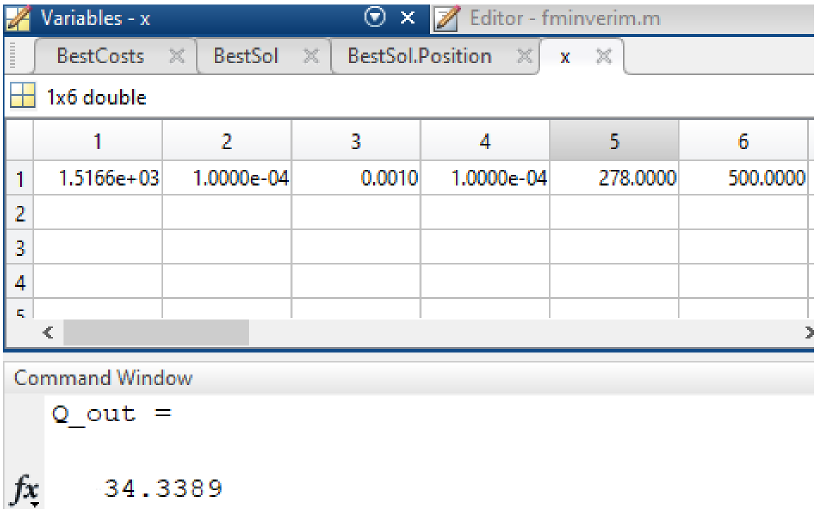

Figure 3.1 Fmincon optimization results for GaSb cell

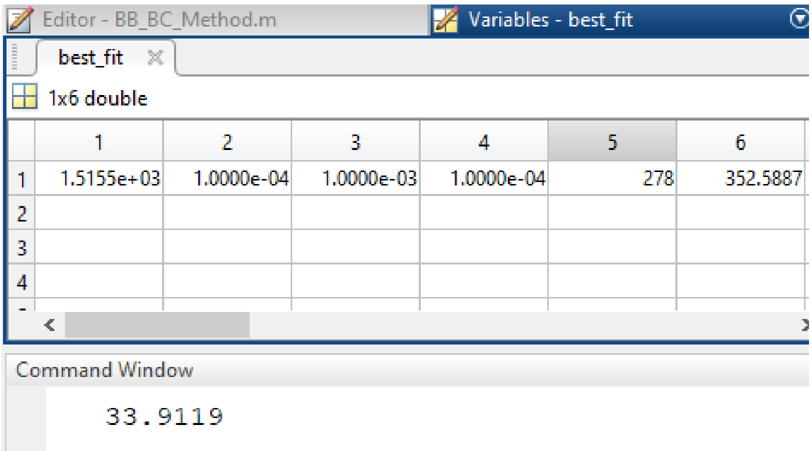

Figure 3.2 Particle swarm optimization results for GaSb cell

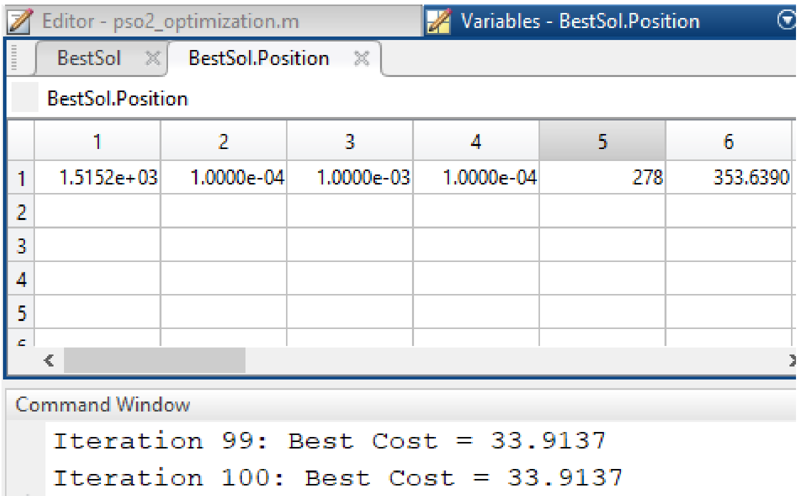

Figure 3.3 Big bang and big crunch optimization results for GaSb cell

3.3. Case - 2: General Optimization of Si Cell

We kept the limits of the parameters in wide ranges to make a general optimization. We obtained the results from the 3 optimization methods we used. We compared these results we obtained. By taking A_{cell} and $A_{emitter}$ values 1 cm^2, we get the power output value per cm^2. These processes are calculated according to the features of Si cell.

Our results showed that thermophotovoltaics using Si cell are less efficient than thermophotovoltaics using GaSb cell under the same conditions in accordance with our literature study.

Table 3.3 Optimization constraints and results for Si cell TPV system

	Fmincon Opt.	**PSO Opt.**	**BB BC Opt.**
$800<X_1$ ($T_{emitter}$) <2000 (K)	**1586.8**	**1586.8**	**1586.2**
$0.0001<X_2$ ($A_{emitter}$) <0.0001 (m^2)	**0.0001**	**0.0001**	**0.0001**
$0.001<X_3$ (Distance) <0.02 (m)	**0.0010**	**0.0010**	**0.0010**
$0.0001<X_4$ (A_{cell}) <0.0001 (m^2)	**0.0001**	**0.0001**	**0.0001**
$278<X_5$ (T_{water}) <350 (K)	**278.00**	**278.00**	**278.00**
$298<X_6$ (T_{cell}) <500 (K)	**364.70**	**364.70**	**363.48**
Q_{output} (W)	**16.617**	**16.376**	**16.375**

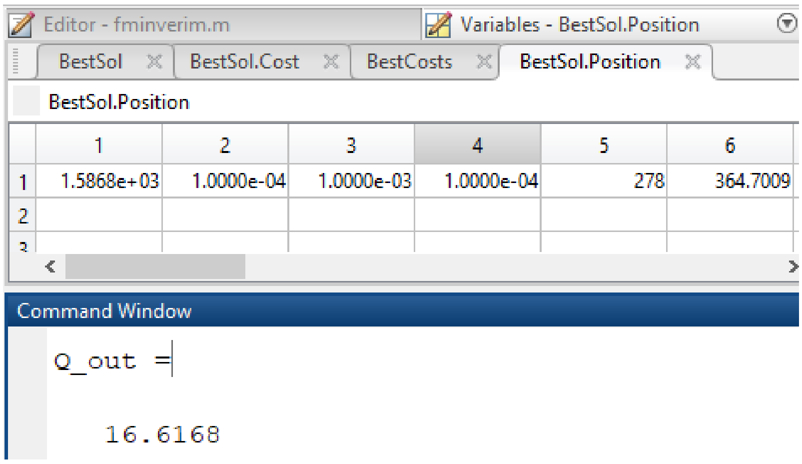

Figure 3.4 Fmincon optimization results for Si cell

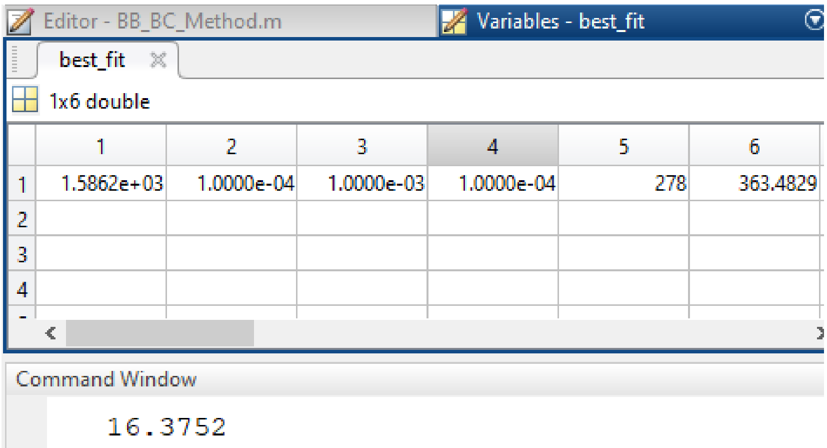

Figure 3.5 Particle swarm optimization results for Si cell

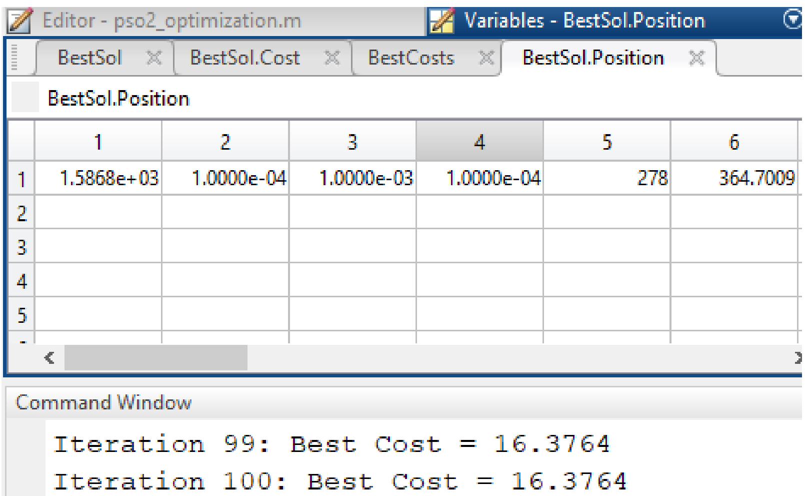

Figure 3.6 Big bang and big crunch optimization results for Si cell

The optimizations we have done above show us the relevant maximum Work Out values when we give a wide range of values to our variables.

Our optimization calculations are made for black body emitter. It was observed that our variables $T_{emitter}$, A_{cell} and $A_{emitter}$ had a direct proportional effect for Work Output. Distance between cell and emitter, T_{water} and T_{cell} have inverse proportion for Work Output. Since the increase in $T_{emitter}$ and $A_{emitter}$ has a decreasing effect on cell efficiency, T_{cell} is restrictive in increasing $T_{emitter}$ and $A_{emitter}$.

3.4. Case 3 : Comparison of Experimantal Results with Our Optimization for GaSb Cell

With reference to the results of an experimental study created similar to the systems we have created in Case 1 and Case 2, we have seen that the results we obtained when we made optimization again were close to the power output values in the experimental study. The results we obtained in Case 3 and Case 4 provided a basis for the accuracy of our work in Case1 and Case 2.

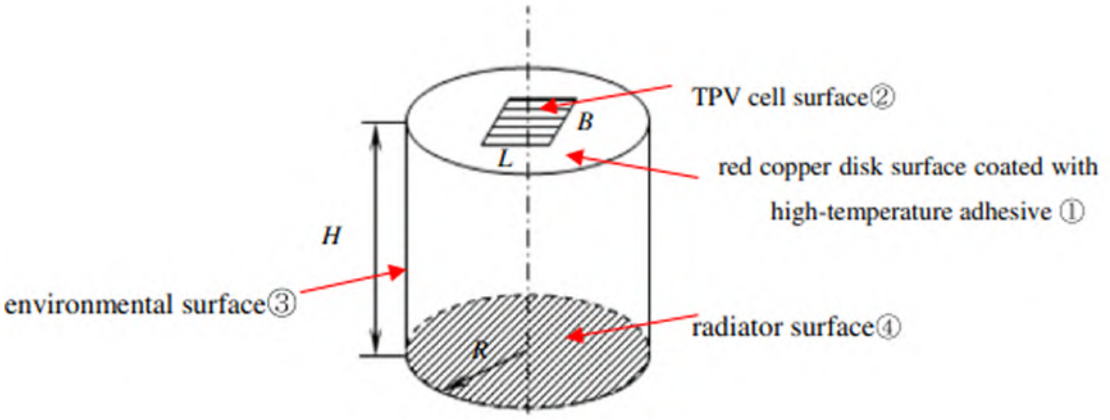

Figure 3.7 Experimental setup schematic view

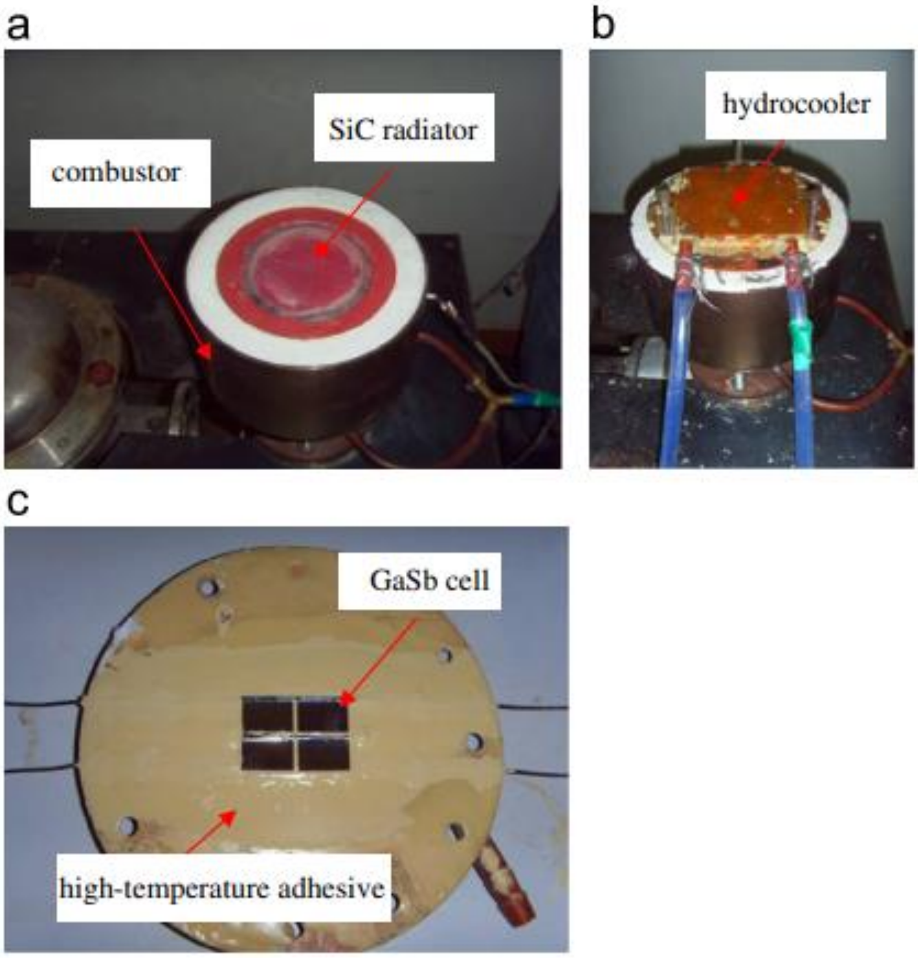

Figure 3.8 Experimental system devices: (a) radiator; (b) hydrocooler; (c) GaSb cell array

Table 3.4 Experimental results for GaSb cell [5]

Experimental results of the output performance of a single GaSb cell.

Combustion power (kW)	Radiator–cell distance (cm)	Maximal radiator surface temperature (K)	High-temperature adhesive temperature (K)	Cell temperature (K)	V_{oc} (V)	I_{sc} (A)	P (W cm^{-2})
2.6	1.1	1150.2	331.3	340.1	0.36	1.99	0.23
	2.0	1145.3	315.9	323.6	0.38	1.90	0.22
	3.0	1144.5	313. 4	320.3	0.39	1.24	0.17
	3.8	1143.0	311.0	317.1	0.40	0.91	0.14
3.2	1.1	1180.0	332.8	343.4	0.35	2.12	0.25
	2.0	1177.2	319.6	330.6	0.37	2.03	0.24
	3.0	1175.4	313.9	323.4	0.38	1.65	0.21
	3.8	1174.3	311.8	320.1	0.39	1.21	0.18
4.2	1.1	1223.2	337.1	350.2	0.34	2.29	0.26

Table 3.5 Our results from our optimization [for shaded experimental values]

	Fmincon Opt.	PSO Opt.	BB BC Opt.
X_1 ($T_{emitter}$)	1180 (K)	1180 (K)	1180 (K)
X_2 ($A_{emitter}$)	0.00073 (m^2)	0.00073 (m^2)	0.00073 (m^2)
X_3 (Distance)	0.011 (m)	0.011 (m)	0.011 (m)
X_4 (A_{cell})	0.000146 (m^2)	0.000146 (m^2)	0.000146 (m^2)
X_5 (T_{water})	300(K)	299 (K)	306(K)
X_6 (T_{cell})	343 (K)	343.4 (K)	343.4 (K)
Q_{output} (W)	0.7826	0.79057	0.7191

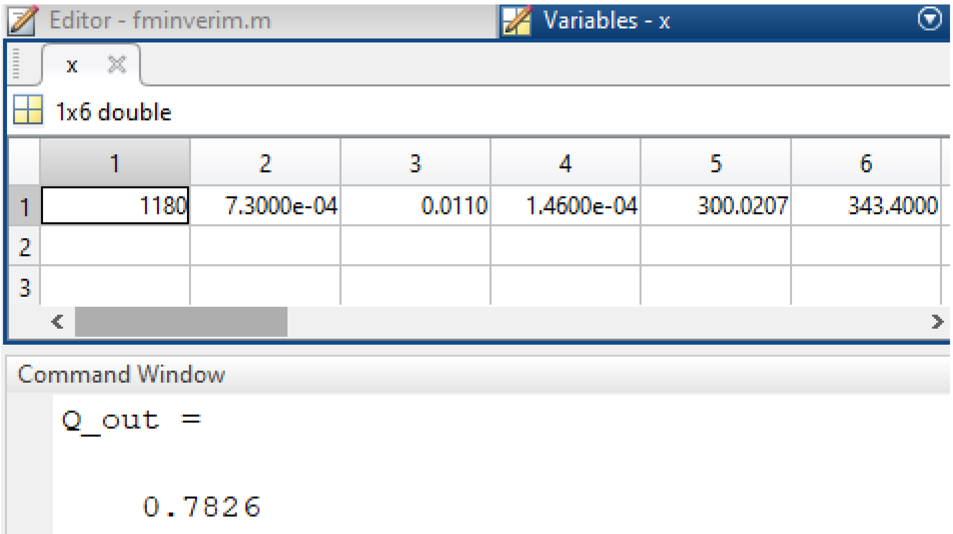

Figure 3.9 Fmincon optimization results for GaSb cell

The result we obtained in optimization is the work output value obtained for the black body emitter and 1.46 cm^2 cell area. In order to compare the experimental results, we assume the emitter emissivity as 0.7 and calculate the Work Output value per cm^2 as follows:

$$Q_{out,exp} = \frac{Q_{out} * \varepsilon^2}{A_{cell}} \ (W/cm^2)$$

$$Q_{out,exp} = \frac{0.7826 * 0.7^2}{1.46} = 0.263 \ (W/cm^2)$$

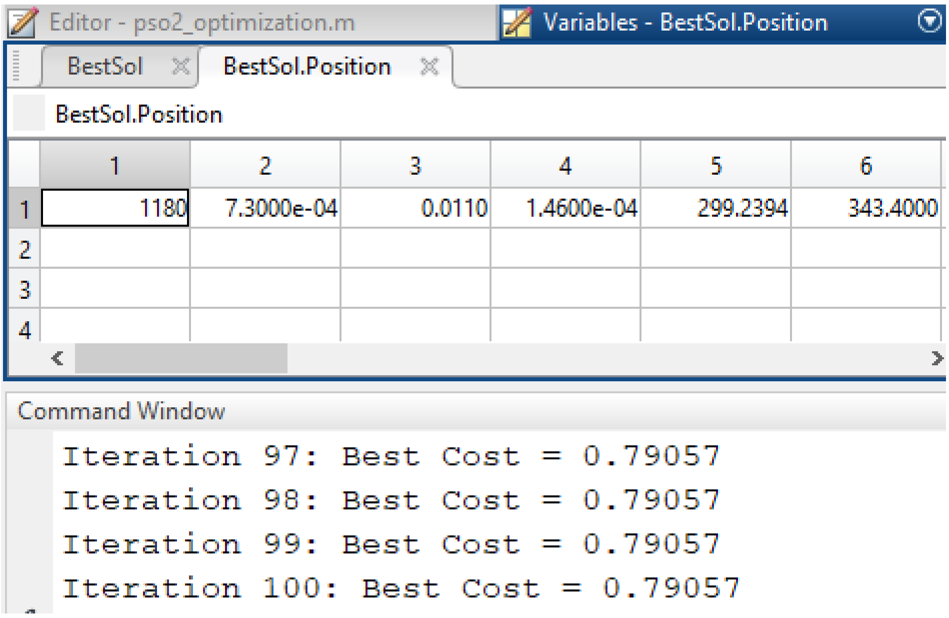

Figure 3.10 Particle swarm optimization results for GaSb cell

The result we obtained in optimization is the work output value obtained for the black body emitter and 1.46 cm^2 cell area. In order to compare the experimental results, we assume the emitter emissivity as 0.7 and calculate the Work Output value per cm^2 as follows:

$$Q_{out,exp} = \frac{Q_{out} * \varepsilon^2}{A_{cell}} \; (W/cm^2)$$

$$Q_{out,exp} = \frac{0.7906 * 0.7^2}{1.46} = 0.265 \; (W/cm^2)$$

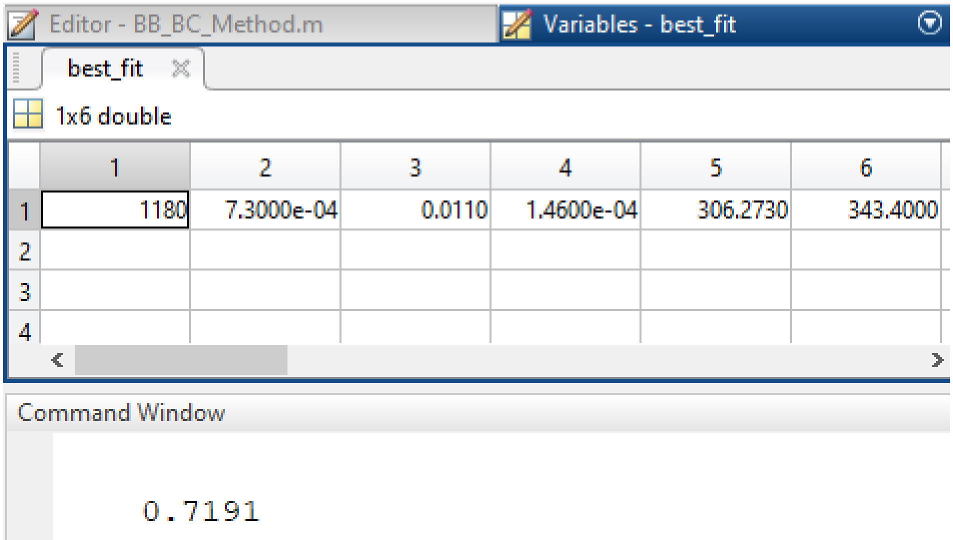

Figure 3.11 Big bang and big crunch optimization results for GaSb cell

The result we obtained in optimization is the work output value obtained for the black body emitter and 1.46 cm^2 cell area. In order to compare the experimental results, we assume the emitter emissivity as 0.7 and calculate the Work Output value per cm^2 as follows:

$$Q_{out,exp} = \frac{Q_{out} * \varepsilon^2}{A_{cell}} \ (W/cm^2)$$

$$Q_{out,exp} = \frac{0.7191 * 0.7^2}{1.46} = 0.241 \ (W/cm^2)$$

As can be seen in the calculations above, our results are consistent and our results from our optimizations match up substantially with experimental results.

3.5. Case 4 : Comparison of Experimantal Results with Our Optimization for Si Cell

Table 3.6 Experimental results for Si cell [5]

Experimental results of the output performance of a single Si cell.

Combustion power (kW)	Radiator–cell distance (cm)	Maximal radiator surface temperature (K)	High-temperature adhesive temperature (K)	Cell temperature (K)	V_{oc} (V)	I_{sc} (A)	P (W cm^{-2})
2.6	1.1	1157.7	353.1	360.4	0.55	0.20	0.03
	2.0	1150.2	336.7	342.1	0.56	0.14	0.02
3.2	1.1	1190.4	358.2	366.9	0.53	0.31	0.05
	2.0	1182.1	340.9	345.4	0.54	0.22	0.04
4.2	1.1	1230.8	362.4	367.6	0.51	0.44	0.06

Table 3.7 Our results from our optimization [for shaded experimental values]

	Fmincon Opt.	**PSO Opt.**	**BB BC Opt.**
X_1 ($T_{emitter}$)	**1230.8 (K)**	**1230.8 (K)**	**1230.8 (K)**
X_2 ($A_{emitter}$)	**0.001 (m^2)**	**0.001 (m^2)**	**0.001 (m^2)**
X_3 (Distance)	**0.011 (m)**	**0.011 (m)**	**0.011 (m)**
X_4 (A_{cell})	**0.0002 (m^2)**	**0.0002 (m^2)**	**0.0002 (m^2)**
X_5 (T_{water})	**300(K)**	**343 (K)**	**342(K)**
X_6 (T_{cell})	**367.6 (K)**	**367.6 (K)**	**367.6 (K)**
Q_{output} (W)	**0.277**	**0.245**	**0.263**

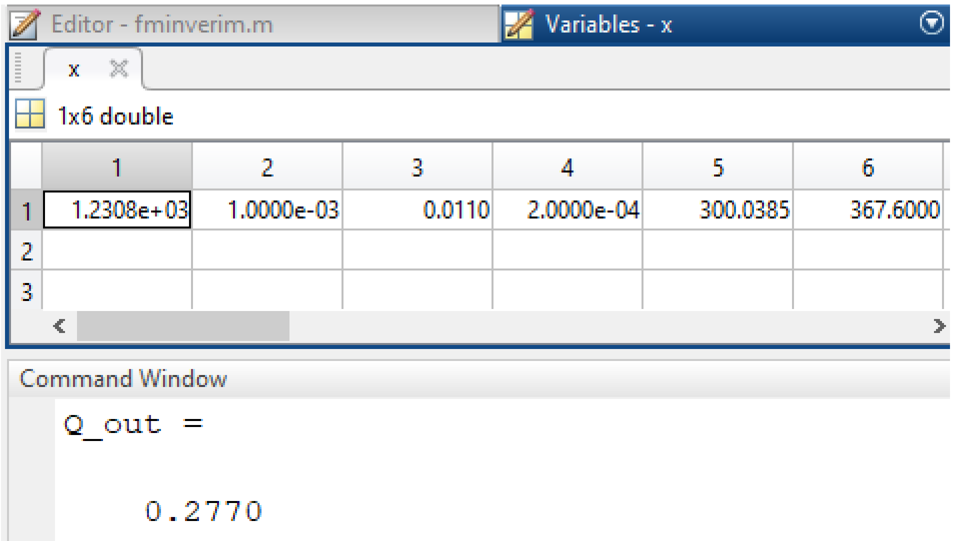

Figure 3.12 Fmincon optimization results for Si cell

The result we obtained in optimization is the work output value obtained for the black body emitter and 2 cm^2 cell area. In order to compare the experimental results, we assume the emitter emissivity as 0.7 and calculate the Work Output value per cm^2 as follows:

$$Q_{out,exp} = \frac{Q_{out} * \varepsilon^2}{A_{cell}} \ (W/cm^2)$$

$$Q_{out,exp} = \frac{0.277 * 0.7^2}{2} = 0.0679 \ (W/cm^2)$$

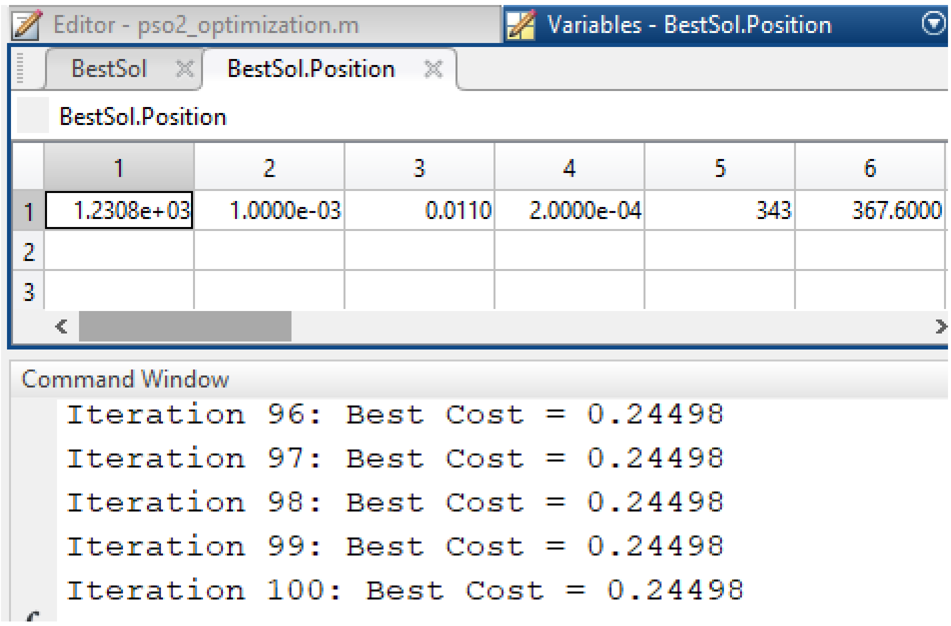

Figure 3.13 Particle swarm optimization results for Si cell

The result we obtained in optimization is the work output value obtained for the black body emitter and 2 cm^2 cell area. In order to compare the experimental results, we assume the emitter emissivity as 0.7 and calculate the Work Output value per cm^2 as follows:

$$Q_{out,exp} = \frac{Q_{out} * \varepsilon^2}{A_{cell}} \; (W/cm^2)$$

$$Q_{out,exp} = \frac{0.245 * 0.7^2}{2} = 0.0600 \; (W/cm^2)$$

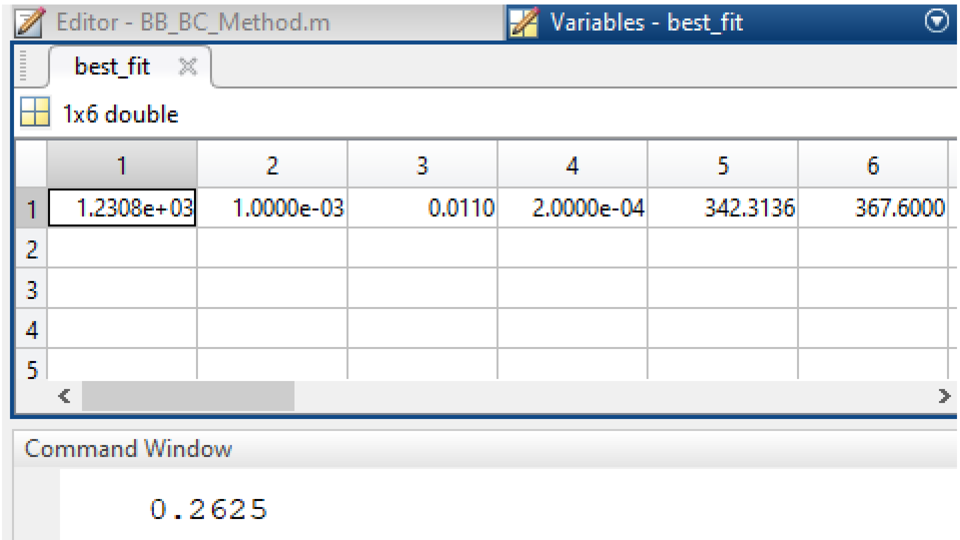

Figure 3.14 Big bang and big crunch optimization results for Si cell

The result we obtained in optimization is the work output value obtained for the black body emitter and 2 cm^2 cell area. In order to compare the experimental results, we assume the emitter emissivity as 0.7 and calculate the Work Output value per cm^2 as follows:

$$Q_{out,exp} = \frac{Q_{out} * \varepsilon^2}{A_{cell}} \; (W/cm^2)$$

$$Q_{out,exp} = \frac{0.263 * 0.7^2}{2} = 0.0644 \; (W/cm^2)$$

As can be seen in the calculations above, our results are consistent and our results from our optimizations match up substantially with experimental results.

4. CONCLUSION

TPV systems are basically systems that convert heat energy into electrical energy. While performing this process, many parameters affect the power output of the system. Some of these parameters also affect each other and show a restrictive feature. The main of these parameters are; $T_{emitter}$, T_{cell}, $A_{emitter}$, A_{cell}, Distance between Cell and Emitter, $T_{coolant}$.

In this study, we calculated the maximum work output values by finding the optimum values of the main parameters affecting the TPV system within the limits determined by the 'Fmincon', 'Particle Swarm' and 'Big Bang and Big Crunch' optimization methods. By comparing our working method with the experimental results, we compared the values we obtained from the optimizations and obtained consistent results. We performed these processes separately for both Si Cell and GaSb Cell TPVs.

When we used 1 mm distance between cell and emitter in the optimizations that we made in wide value ranges, we observed the maximum work output value as 32 W/cm^2 in TPV systems using Si Cell and black body emitter, and 32 W/cm^2 in TPV systems using GaSb Cell and black body emitter.

In the cooling heat transfer calculations, we used water as the refrigerant and took the flow temperature of the water as 5 degrees and the velocity of the fluid as 3 m/s and Pr_{water}, V, ρ , μ and k_{water} values were assumed to be fixed according to the values of the water at 5 degrees. Cell thickness is assumed to be 0.04 mm. In our TPV system, it is assumed to be vacuumed between cell and emitter. The more truthful these assumptions we have taken, the better the results will be.

Our optimization work can contribute to the improvement of existing TPV systems. The effects of the basic parameters affecting TPV systems on Work Output and also their relationships among themselves have been revealed. With the help of these optimizations, a control mechanism can be created that will allow the TPV system to produce the highest possible Work Output value according to changing conditions.

REFERENCES

1. One-Chip Near-Field Thermophotovoltaic Device Integrating a Thin-Film Thermal Emitter and Photovoltaic Cell

Takuya Inoue, Takaaki Koyama, Dongyeon Daniel Kang, Keisuke Ikeda, Takashi Asano, and Susumu Noda
Nano Letters 2019 *19* (6), 3948-3952
DOI: 10.1021/acs.nanolett.9b01234

2. https://www.cs.cinvestav.mx/~constraint/papers/cagnina08a.pdf

3. X. S. Yang, "Optimization and Metaheuristic Algorithms in Engineering," in: Metaheuristics in Water, Geotechnical and Transport Engineering (Eds. X. S. Yang, A. H. Gandomi, S. Talatahari, A. H. Alavi), Elsevier, (2013), pp. 1-23.

4. J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, 1995, pp. 1942-1948 vol.4

5. Experimental analysis of cell output performance for a TPV system Xi Wu, Hong Ye, Jianxiang Wang, August 2011

6. Thermal and Performance Analysis of a Photovoltaic Module with an Integrated Energy Storage System Manel Hammami, Simone Torretti, Francesco Grimaccia Iand Gabriele Grandi, 25 October 2017