VICTORIA UNIVERSITY OF WELLINGTON TE HERENGA WAKA NEW ZEALAND
Archives of Sustainable Energy Systems
https://www.wgtn.ac.nz/sustainable-energy-systems

Assessing the impact of solar PV on vegetation growth through ground sunlight distribution at a solar farm in Aotearoa New Zealand Supplementary Material

PPFD required for various plants

Plant	Genus	PPFD (μmol/m²/s)	DLI (mol/m²/d)
African Violets	Streptocarpus	70 – 300	4 – 14
Aglaonema	Aglaonema	20 – 40	4 – 14
Alocasia	Alocasia	80 – 500	4 – 14
Aloe	Aloe	80 – 160	4 – 14
Amaryllis	Amaryllis	150 – 300	4 – 14
Anthurium	Anthurium	80 – 400	4 – 14
Aspidistra	Aspidistra	40 – 400	4 – 18
Avocado	Persea americana	400 – 650	18 – 24
Basil	Ocimum basilicum	220 – 500	12 – 26
Blackberry	Rubus	200 – 300	8 – 14
Blueberry	Vaccinium	400 – 700	14 – 24
Broccoli	Brassica oleracea var. italica	300 – 700	15 – 35
Bromeliad	Bromeliaceae	80 – 600	4 – 14
Calathea	Goeppertia	80 – 400	4 – 15
Cannabis (Flowering)	Cannabis	500 – 1050	30 – 40
Cannabis (Seedling)	Cannabis	100 – 300	12 – 16
Cannabis (Vegetative)	Cannabis	250 – 600	20 – 45
Chrysanthemums	Chrysanthemum	200 – 300	10 – 14
Cilantro	Coriandrum sativum	300 – 600	20 – 28
Citrus (Various)	Citrus	300 – 600	20 – 28
Croton	Croton	80 – 160	4 – 16
Cucumbers	Cucurbitaceae	300 – 600	20 – 30
Cucumbers (Seedling)	Cucurbitaceae	100 – 300	5 – 15
Culinary Herbs (Small)	Various	150 – 250	10 – 12
Cyclamen	Primulaceae	70 – 150	6 – 8
Desert Cacti	Various	500 – 2000	9 – 30
Dill	Anethum graveolens	300 – 600	20 – 28
Dracaena	Dracaena	20 – 40	4 – 14
Dragon Tree	Dracaena	20 – 40	4 – 14
Dwarf Banana	Musa acuminata	100 – 350	4 – 14
Eggplants	Solanaceae	300 – 600	20 – 30
English Ivy	Hedera	40 – 80	4 – 8
Ferns	Pteridaceae	80 – 120	4 – 6
Ferns	Asplenium	20 – 40	5 – 15
Ficus	Ficus	80 – 160	6 – 12
Fuchsia	Fuchsia	150 – 250	10 – 12
Garden Lettuce	Lactuca	250 – 350	14 – 16
Grapefruit	Citrus	500 – 700	22 – 30
Hibiscus	Hibiscus	80 – 500	4 – 14
Hoya	Hoya	40 – 400	4 – 12
Jasmine	Stephanotis	200 – 400	5 – 16
Kalanchoe	Kalanchoe	60 – 120	9 – 30
Kumquat	Citrus	500 – 700	22 – 30
Lemon	Citrus	300 – 600	21 – 28
Lime	Citrus	300 – 600	21 – 28
Mint	Mentha	200 – 400	10 – 20
Money Tree	Pachira	40 – 300	4 – 14
Monstera	Monstera	80 – 500	4 – 12
Olive	Olea	400 – 1000	18 – 40
Orange	Citrus	500 – 700	22 – 30
Orchids (High-Light)	Orchidaceae	150 – 350	8 – 18
Orchids (Low-Light)	Orchidaceae	40 – 80	4 – 6
Orchids (Moderate-Light)	Orchidaceae	80 – 150	4 – 8
Oregano	Origanum vulgare	300 – 600	20 – 28
Oxalis	Oxalis	150 – 250	2 – 10
Palm Trees	Various	80 – 1200	10 – 40
Parsley	Petroselinum crispum	200 – 400	10 – 20
Peace Lily	Spathiphyllum	20 – 40	4 – 14
Peach	Prunus	200 – 300	8 – 14
Peas	Pisum sativum	150 – 300	9 – 12
Peperomia	Peperomia	40 – 300	4 – 12
Peppers	Capsicum annuum	300 – 600	20 – 30
Peppers (Seedling)	Capsicum annuum	150 – 350	8 – 18
Petunias	Petunia	300 – 400	16 – 18
Philodendron	Philodendron	40 – 300	4 – 14
Pineapple	Ananas	350 – 700	15 – 30
Polka Dot Plant	Hypoestes	40 – 80	4 – 14
Pothos	Epipremnum	40 – 600	4 – 14
Pumpkins	Cucurbita	300 – 650	18 – 28
Rosemary	Rosmarinus officinalis	300 – 600	20 – 28
Roses	Rosa	350 – 450	18 – 22
Sage	Salvia officinalis	200 – 400	10 – 20
Schefflera	Schefflera	40 – 800	7 – 30
Seedlings	Various	70 – 150	6 – 8
Snake Plant	Dracaena	40 – 600	4 – 14
Spider Plant	Chlorophytum	40 – 80	4 – 14
Strawberry	Fragaria	400 – 600	17 – 28
Succulents	Various	75 – 150	3 – 6
Syngonium	Syngonium	40 – 600	4 – 14
Tangerine	Citrus	500 – 700	22 – 30
Thyme	Thymus vulgaris	300 – 600	20 – 28
Tomatoes	Solanum lycopersicum	350 – 800	22 – 30
Tomatoes (Seedling)	Solanum lycopersicum	150 – 350	8 – 18
Zucchini	Cucurbita pepo	400 – 650	22 – 28
ZZ Plant	Zamioculcas	200 – 400	2 – 10
ZZ Plant (Propagation)	Zamioculcas	20 – 40	1 – 3

(Source: Photone, 2025)

Code used to plot the GHIground and PPFD heat maps

```
import pandas as pd
import numpy as np
import geopandas as gpd
import matplotlib.pyplot as plt
from pvlib.location import Location
from shapely.ops import unary_union
from shapely import affinity
from shapely.ops import unary_union
import matplotlib as mpl
from shapely.geometry import Polygon, Point
import geopandas as gpd
from mpl_toolkits.axes_grid1 import make_axes_locatable
from matplotlib import cm, colors
import os
```

```
# -- Define function to divide panel into 1m x 1m cells ---
def divide_panel_into_cells(panel_bounds, cell_size=1.0):
    minx, miny, maxx, maxy = panel_bounds
    cells = []
    x = minx
    while x < maxx:
        y = miny
        while y < maxy:
            cell = Polygon([
                (x, y),
                (min(x + cell_size, maxx), y),
                (min(x + cell_size, maxx), min(y + cell_size, maxy)),
                (x, min(y + cell_size, maxy))
            ])
            cells.append(cell)
            y += cell_size
        x += cell_size
    return cells

# --- Step 2: Define function to divide study area into 1m x 1m cells ---
def divide_study_area_into_cells(area_bounds, cell_size=1.0):
    minx, miny, maxx, maxy = area_bounds
    cells = []
    x = minx
    while x < maxx:
        y = miny
        while y < maxy:
            cell = Polygon([
                (x, y),
                (min(x + cell_size, maxx), y),
                (min(x + cell_size, maxx), min(y + cell_size, maxy)),
                (x, min(y + cell_size, maxy))
            ])
            cells.append(cell)
            y += cell_size
        x += cell_size
    return cells
```

```
# --- Loop over elevations and azimuths ---  TRY2
for elevation in range(90, -1, -10):
    shading_factors[elevation] = {}
    for azimuth in range(-180, 181, 20):
        shading_table = []

        if elevation == 0:
            for idx, cell in enumerate(cells):
                shading_table.append({
                    'cell_id': idx,
                    'cell_geometry': cell,
                    'shading_factor': 1.0  # assume fully shaded at sunrise/sunset
                })
        else:
            # ---  Project all panel shadows ---
            panel_shadow_polygons = []
            for panel_cell in panel_cells:
                panel_vertices = []
                for (x, y) in panel_cell.exterior.coords:
                    y_ratio = (y - panel_bounds[1]) / (panel_bounds[3] - panel_bounds[1])
                    z = 0.9 + (2.6 - 0.9) * y_ratio
                    panel_vertices.append((x, y, z))
                shadow = project_shadow(panel_vertices, elevation, azimuth)
                panel_shadow_polygons.append(shadow)

            # Combine all shadows into one geometry
            combined_shadow = unary_union(panel_shadow_polygons)

            # --- Check how much each cell is shaded ---
            for idx, cell in enumerate(cells):
                intersection_area = combined_shadow.intersection(cell).area
                shading_factor = intersection_area / cell.area if cell.area > 0 else 0.0

                shading_table.append({
                    'cell_id': idx,
                    'cell_geometry': cell,
                    'shading_factor': shading_factor
                })

        shading_factors[elevation][azimuth] = shading_table
```

```
# --- Save all shading factors into a multi-indexed CSV ---
records = []
for elevation, azimuth_dict in shading_factors.items():
    for azimuth, table in azimuth_dict.items():
        for entry in table:
            records.append({
                'elevation': elevation,
                'azimuth': azimuth,
                'cell_id': entry['cell_id'],
                'shading_factor': entry['shading_factor']
            })

shading_df = pd.DataFrame(records)
shading_df.to_csv('shading_factors.csv', index=False)

shading_df.head()
```

```
#  Load weather file and match shading factors
weather_df = pd.read_csv('SolaRGIS-Maraetai_Dam_2018.csv', sep=';')
weather_df['Datetime'] = pd.to_datetime(weather_df['Date'] + ' ' + weather_df['Time'], format='%d.%m.%Y %H:%M')
weather_df.set_index('Datetime', inplace=True)
weather_df.drop(columns=['Date', 'Time', 'PVOUT', 'GTI', 'flagR', 'TEMP', 'WS', 'WD', 'RH', 'AP', 'PWAT', 'PREC', 'WG'], inplace=True)
weather_df.rename(columns={'SE': 'Elevation', 'SA': 'Azimuth'}, inplace=True)
weather_df.index = weather_df.index.tz_localize('Europe/Bratislava', nonexistent='shift_forward', ambiguous=True).tz_convert('Pacific/Auckland')
```

```
    ghi_ground_dict[f'Cell_{cell_id}'] = cell_GHI_ground


final_df = pd.DataFrame(ghi_ground_dict, index=weather_subset.index)
```

Heat Map with standardised legend

```
# STore in file
output_dir = "plots"
os.makedirs(output_dir, exist_ok=True)

final_df['date'] = final_df.index.date
daily_avg = final_df.drop(columns=['date', 'Season'], errors='ignore').groupby(final_df['date']).mean()

# Compute global min and max for standardized color scale
global_min = daily_avg.min().min()
global_max = daily_avg.max().max()


# Loop through each seasonal date
for date in daily_avg.index:
    cell_values = daily_avg.loc[date].values  # One value per cell
    gdf = gpd.GeoDataFrame({'geometry': cells, 'GHI_ground': cell_values})

    fig, ax = plt.subplots(figsize=(8, 6))
    gdf.plot(column='GHI_ground', cmap='plasma', legend=True, ax=ax,
             vmin=global_min, vmax=global_max)

    ax.set_aspect('equal')  # maintain spatial scale

    ax.set_title(f'Daily Avg GHI_ground\n{date}', fontsize=14)
    ax.set_xlabel('Row Length (m)')
    ax.xaxis.set_label_position('top')
    ax.xaxis.tick_top()
    ax.invert_xaxis()

    ax.set_ylabel('Row Width and Pitch (m)')
    ax.invert_yaxis()

    # Save the figure
    filename = os.path.join(output_dir, f'GHI_ground_standardised_{date.strftime("%Y-%m-%d")}.png')
    plt.tight_layout(pad=0)  # minimize internal spacing
    plt.savefig(filename, dpi=300, bbox_inches='tight', pad_inches=0)
    plt.show()
```

PPFD Heat Maps

```
# Ensure output directory exists
output_dir = "ppfd_plots"
os.makedirs(output_dir, exist_ok=True)

# Scale GHI_ground to PPFD (µmol m⁻² s⁻¹)
ppfd_df = daily_avg * 2.02  # Element-wise conversion
global_min = ppfd_df.min().min()
global_max = ppfd_df.max().max()

# Define fixed axes limits
x_min, x_max = -7.5, 7.5
y_min, y_max = 0.0, 7.5

# Loop through each seasonal date
for date in ppfd_df.index:
    cell_values = ppfd_df.loc[date].values
    gdf = gpd.GeoDataFrame({'geometry': cells, 'PPFD': cell_values})

    fig, ax = plt.subplots(figsize=(8, 6))
    gdf.plot(column='PPFD', cmap='YlGn', legend=True, ax=ax,
             vmin=global_min, vmax=global_max)

    ax.set_xlim(x_min, x_max)
    ax.set_ylim(y_min, y_max)
    ax.set_aspect('equal')

    ax.set_title(f'Daily Avg PPFD\n{date}', fontsize=14)
    ax.set_xlabel('Row Length (m)')
    ax.xaxis.set_label_position('top')
    ax.xaxis.tick_top()
    ax.invert_xaxis()

    ax.set_ylabel('Row Width and Pitch (m)')
    ax.invert_yaxis()

    # Optional: Label the colorbar
    cbar = ax.get_figure().get_axes()[1]
    cbar.set_ylabel("PPFD (µmol m⁻² s⁻¹)", fontsize=12)

    # Save the figure
    filename = os.path.join(output_dir, f'PPFD_standardised_{date.strftime("%Y-%m-%d")}.png')
    plt.savefig(filename, dpi=300)
    plt.show()
```