

Projeto de Graduação



30/06/2023

Régressão não-linear de irradiância solar em módulos fotovoltaicos monofaciais fixos que utilizam os modelos de transposição de Perez e de Hay-Davies

Rafael Vilela Santa Rosa



www.ele.puc-rio.br

**Régressão não-linear de irradiância solar em
módulos fotovoltaicos monofaciais fixos que
utilizam os modelos de transposição de Perez
e de Hay-Davies**

Aluno(s): Rafael Vilela Santa Rosa

Orientador(es): Carlos Hall

Rodrigo Calili

Agradecimentos

Eu agradeço imensamente a todos pela conclusão dessa jornada, em especial a Deus, minha família, meus amigos, colegas de PUC e RioBotz, além das pessoas que trabalham e trabalharam comigo na Localpower por mais de 2 anos que – diretamente ou indiretamente – possibilitaram a entrega do presente trabalho de conclusão de curso. O mérito é todo de vocês e não meu, porque sozinho não seria possível concluir minha graduação.

Além disso, não poderia deixar de agradecer a todas as pessoas que há décadas ajudaram a moldar o caminho para a energia solar ser viável e amplamente utilizada atualmente. Imagino que foram anos de trabalho intenso a fim de possibilitar o avanço nessa área e – sem licença poética – ajudarem, assim como outros profissionais de outras áreas, a melhorar o mundo como conhecemos.

“Quanto ao mais, irmãos, tudo o que é verdadeiro, tudo o que é honesto, tudo o que é justo, tudo o que é puro, tudo o que é amável, tudo o que é de boa fama, se há alguma virtude, e se há algum louvor, nisso pensai.”

Resumo

O seu objetivo é propor a aplicação de regressão não-linear em python utilizando o método de Levenberg-Marquardt para estimar os valores de irradiação em módulos fotovoltaicos monofaciais e fixos, que são a maioria no planeta. O problema da simulação de módulos fotovoltaicos, de uma forma geral, são os dados imprecisos meteorológicos com outliers ocasionando discrepâncias na simulação de geração de energia em sistemas fotovoltaicos. A partir do método aplicado para o valor de irradiação no módulo é possível tratar os valores muito afastados da tendência de um comportamento senoidal da irradiação solar na Terra, sejam valores altos ou muito baixos. As referências utilizadas foram as simulações no software PVsyst (considerando o Modelo de Perez pois não possui o Hay-Davies) e o HelioScope (Perez e Hay-Davies), entretanto a intenção é que seja utilizado de uma forma ampla por qualquer outro software ou de dados locais. A base meteorológica utilizada foi a da estação no Aeroporto Internacional de Piedmont Triad, na Carolina do Norte, que é classificada como “classe I” pela NREL e recomendada pela biblioteca pvlib para simulações.

Palavras-chave: **energia solar; Levenberg-Marquardt; regressão não-linear; módulos fotovoltaicos; python**

Non-linear regression of solar irradiance in fixed monofacial photovoltaic modules using Perez and Hay-Davies transposition models

Abstract

Its objective is to propose the application of non-linear regression in python using the Levenberg-Marquardt method to estimate the irradiance values in fixed monofacial photovoltaic modules, which are the majority on the planet. The problem with simulating photovoltaic modules, in general, is inaccurate meteorological data with outliers causing discrepancies in the simulation of energy generation in photovoltaic systems. From the method applied to the irradiance value in the module, it is possible to treat values that are very far from the tendency of a sinusoidal behavior of solar irradiation on Earth, whether high or very low values. The references used were the simulations in the PVsyst software (considering the Perez Model since it does not have Hay-Davies) and the HelioScope (Perez and Hay-Davies), however the intention is that it could be used in general by any other software or with local data. The meteorological base used was the station at Piedmont Triad International Airport, in North Carolina, which is classified as "Class I" by NREL and recommended by the pvlib library for simulations.

Keywords: solar energy; Levenberg-Marquardt; non-linear regression; photovoltaic modules; python

Sumário

1. Introdução	14
1.1 Objetivo.....	14
1.2 Breve história da energia solar	15
1.2.1 Energia solar no Brasil	15
1.3 Irradiância no coletor ou no plano (plane of array)	18
2. Ondas eletromagnéticas e fóttons.....	19
2.1 Básico da Teoria Quântica	20
2.2 Espectro eletromagnético do Sol	20
2.3 Constante Solar	21
3. Efeitos advindos das ondas eletromagnéticas.....	22
3.1 Efeito fotoelétrico	22
3.2 Efeito fotovoltaico	23
4. Módulos fotovoltaicos e energia solar	25
4.1 Concepção do módulo fotovoltaico	25
4.2 Modelo de 1 diodo e suas variações.....	26
4.3 Classificação quanto aos testes em STC	27
4.3.1 Irradiância	28
4.3.2 Temperatura	28
4.3.3 Massa de ar	28
4.3.4 Outros parâmetros	28
5. Componentes da irradiação	29
5.1 GHI	29
5.2 DNI	30
5.2.1 DNI Extraterrestre	31
5.3 DHI	31
5.4 Modelos de Transposição.....	32
6. Albedo	33
6.1 Albedo como constante	33

6.2 Albedo como variável	33
7. Principais modelos de transposição da irradiação.....	35
7.1 Modelo de Hay-Davies	35
7.2 Modelo de Perez.....	36
7.3 Modelo de Liu & Jordan.....	39
8. Softwares de simulação	41
8.1 PVsyst	41
8.2 SAM	42
8.3 HelioScope	42
9. Incerteza associada a meteorologia	44
9.1 Fenômenos cíclicos.....	45
9.2 Eventos desconhecidos	46
9.3 Equipamentos para medição.....	47
10. Aprofundamento na regressão não-linear	50
10.1 Regressão linear	50
10.2 Regressão não-linear	51
11. Algoritmo para modelagem de outliers de irradiância solar	52
11.1 Regressão não-linear com o Algoritmo Levenberg–Marquardt	52
11.2 Medidas estatísticas e de análise de sinal	53
11.2.1 Assimetria.....	53
11.2.2 Envelope.....	53
11.2.3 Curtose.....	54
11.2.4 Coeficiente de variação	54
11.3 Código em python do algoritmo de tratamento de outliers.....	55
12. Resultados.....	64
12.1 Resultados do algoritmo no Modelo de Perez.....	64
12.2 Resultados do algoritmo no Modelo de Hay-Davies	72
12.3 Resultados das medidas estatísticas e de análise de sinal	78
13. Conclusão.....	79
14. Referências.....	86
APÊNDICE A - FUNÇÕES ADAPTADAS DO PVLIB (Perez)	90

APÊNDICE B - FUNÇÕES ADAPTADAS DO PVLIB (Hay-Davies).....	102
ANEXO A – BIBLIOTECA PVLIB E SUAS PRINCIPAIS FUNÇÕES UTILIZADAS	113
ANEXO B – BIBLIOTECA SCIPY E SUAS PRINCIPAIS FUNÇÕES UTILIZADAS	118

Lista de figuras

Figura 1- Infográfico 1: Panorama da solar fotovoltaica no Brasil e no mundo (ABSOLAR/ANEEL, 2023)	16
Figura 2 - Infográfico 2: Panorama da solar fotovoltaica no Brasil e no mundo (ABSOLAR/ANEEL, 2023)	17
Figura 3 - Faixa espectral de radiação (Encyclopaedia Britannica, 2023)	19
Figura 4 - Representação gráfica da onda eletromagnética (Encyclopaedia Britannica, 2023)	20
Figura 5 – Experimento do Efeito Fotoelétrico (UFRGS, s.d.).....	22
Figura 6 - Relação energia e trabalho do metal (UFRGS, s.d.).....	23
Figura 7 – Efeito Fotovoltaico (Energy Education (University of Calgary), s.d.).....	24
Figura 8 – Resposta espectral da célula de silício (PVEducation, s.d.).....	26
Figura 9 - Modelo de 1 diodo (PVPMC SANDIA, s.d.)	26
Figura 10 - Modelo de 1 diodo do PVsyst (PVPMC SANDIA, s.d.)	27
Figura 11 – Ângulos e definições geométricas (Atlas brasileiro de energia solar 2º Ed, 2017)	29
Figura 12 – Mapa da GHI no Brasil (Atlas brasileiro de energia solar 2º Ed, 2017).....	30
Figura 13 – Lei da reflexão e refração (Encyclopaedia Britannica, 2023)	31
Figura 14 – Relação entre os Inputs e Outputs do Modelo de Perez (PEREZ et al., 1990).....	36
Figura 15 - Diagrama de perdas do PVsyst (PVsyst7, s.d.)	41
Figura 16 - Interface do SAM (SAM General Description, 2017).....	42
Figura 17 - Arquivo meteorológico de satélite (HELIOSCOPE, 2023)	45
Figura 18 - Anomalia de temperatura devido ao El Niño em 2023 (INPE, 2023)	46
Figura 19 – Foto de uma das estações de coleta de dados de radiação (Atlas brasileiro de energia solar 2º Ed, 2017)	47
Figura 20 – Estações meteorológicas automáticas (EMA's) (Atlas brasileiro de energia solar 2º Ed, 2017)	47
Figura 21 – Piranômetro (Atlas brasileiro de energia solar 2º Ed, 2017)	48
Figura 22 - Piranômetro de fotodiodo e seu leitor (Autoria própria, 2023)	48
Figura 23 – Pireliômetro (Atlas brasileiro de energia solar 2º Ed, 2017)	49
Figura 24 - Bibliotecas em python utilizadas para o código	55
Figura 25 - Definições dos parâmetros.....	56
Figura 26 - Dataframe com valores de irradiância	57
Figura 27 - Cálculo de algumas métricas relevantes	58

Figura 28 - Modelagem da função de tendência	59
Figura 29 - Tratamento de outliers (Parte I)	60
Figura 30 - Tratamento de outliers (parte II)	61
Figura 31 - Comparação entre os valores com e sem correção de outliers	62
Figura 32 - Análise de métricas relevantes após o tratamento de outliers.....	63
Figura 33 - Irradiância no plane of array ao longo do ano sem tratamento de outliers	65
Figura 34 - Irradiância no plane of array em Junho sem tratamento de outliers	65
Figura 35 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária	66
Figura 36 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária - com limitação de potência pelo percentil 98	66
Figura 37 - Irradiância no plane of array ao longo do ano com tratamento de outliers	67
Figura 38 - Irradiância no plane of array em Junho com tratamento de outliers	67
Figura 39 - Regressão não-linear senoidal das curvas diárias de irradiância	68
Figura 40 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária - com tratamento de outliers	68
Figura 41 - Comparação dos valores mensais de energia antes e após a regressão - Perez/HelioScope	69
Figura 42 - Comparação dos valores mensais de energia antes e após a regressão - Perez/PVsyst	69
Figura 43 - Condições de simulação do PVsyst (Perez)	72
Figura 44 - Irradiância no plane of array ao longo do ano sem tratamento de outliers	73
Figura 45 - Irradiância no plane of array em Junho sem tratamento de outliers	73
Figura 46 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária	74
Figura 47 - Comparação dos valores mensais de energia antes e após a regressão - Hay-Davies/HelioScope	74
Figura 48 - Irradiância no plane of array em Junho com tratamento de outliers	75
Figura 49 - Regressão não-linear senoidal das curvas diárias de irradiância	75
Figura 50 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária - com tratamento de outliers	76
Figura 51 - Comparação dos valores mensais de energia antes e após a regressão - Hay-Davies/HelioScope	76
Figura 52 - Irradiância no plane of array ao longo do ano com tratamento de outliers pela energia diária	81
Figura 53 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária	82
Figura 54 - Irradiância no plane of array em Junho com tratamento de outliers pela energia diária	82
Figura 55 - Comparação dos valores mensais de energia antes e após a regressão - Perez/HelioScope	83

Figura 56 - Comparação dos valores mensais de energia antes e após a regressão - Perez/PVsyst 83

Figura 57 - Comparação dos valores mensais de energia antes e após a regressão - Hay-Davies/HelioScope 84

Lista de tabelas

Tabela 1 - Valores do coeficiente de Albedo (Adaptado do PVsyst, s.d.)	33
Tabela 2 - RMSE e MBE do Modelo de Perez, Hay e Isotrópico (PEREZ et al.,1990).....	37
Tabela 3 - Limites para o coeficiente de clareza (Adaptado, PEREZ et al., 1990)	38
Tabela 4 - Coeficientes de clareza do Modelo de Perez (PEREZ et al., 1990)	38
Tabela 5 - Perdas típicas no HelioScope (HelioScope 101, 2023)	43
Tabela 6 - Classificação da estação de Piedmont Triad (NREL, 2008)	52
Tabela 7 - Incerteza da estação de Piedmont Triad ao longo dos anos (NREL, 2012)	52
Tabela 8 - POA simulado no HelioScope (Perez)	70
Tabela 9 - Condições de simulação HelioScope (Perez)	70
Tabela 10 - POA simulado no PVsyst (Perez)	71
Tabela 11 - POA simulado no HelioScope (Hay)	77
Tabela 12 - Condições de simulação HelioScope (Hay)	77
Tabela 13 - Características das curvas de irradiância antes do tratamento de outliers.....	78
Tabela 14 - Características das curvas de irradiância após o tratamento de outliers.....	78
Tabela 15 - Comparação anual final entre os cenários (valor do software dividido pelo valor simulado)	85

Lista de quadros

Quadro 1 - Condições de simulação PVsyst (Perez)	71
Quadro 2 - Estações meteorológicas utilizadas no Modelo de Perez (PEREZ, 1990)	80
Quadro 3 - Conjuntos meteorológicos da NREL (NREL, s.d.)	81

1. Introdução

É notório que, de acordo com as Leis da Termodinâmica, o ser humano não descobriu como criar energia antigamente, porque é impossível criá-la a partir do nada. Pode-se dizer que a etimologia da palavra “energia” remete a capacidade de um elemento ou sistema físico de realizar trabalho, porém para ser possível manipular a energia (seja elétrica, térmica, cinética ou potencial gravitacional) é necessário que aconteça alguma transformação de massa em energia ou de alguma forma de energia para outra.

Há diversos meios disponíveis para a transformação da energia, sendo uma das mais populares e em voga a energia fotovoltaica. Há uma dificuldade, assim como na energia eólica, de previsibilidade de geração dos painéis fotovoltaicos, sejam eles individuais e pequenos, sejam os painéis a nível comercial que são utilizados em imensas usinas fotovoltaicas de centenas de MW ou até mesmo alguns GW de potência instalada. Uma incerteza que acompanha a simulação da geração fotovoltaica é a incerteza associada a algumas estações meteorológicas baseadas em solo que, por erros em instrumentos, falta de dados ou manuseio errado dos equipamentos podem fornecer dados incongruentes de geração que, ao serem associados aos modelos matemáticos, especialmente os de transposição de irradiação (de céu difuso), fornecem valores pouco confiáveis.

1.1 Objetivo

O princípio da conservação de energia de Einstein dita a equivalência massa-energia, crucial para o desenvolvimento da teoria da relatividade geral junto a outros cientistas como Max Planck, que descobriu que a energia é quantizada e Paul Dirac, responsável pela explicação do comportamento dos elétrons em termos relativísticos. A partir deles e demais cientistas nós conseguimos compreender melhor a natureza física e como manipulá-la a fim de extrair benefícios, porém a maioria morreu antes de vislumbrar a atual transição da matriz energética.

Neste contexto, as primeiras experiências, infelizmente, para a geração de energia elétrica em larga escala foram a partir de compostos – denominados posteriormente de combustíveis - poluentes ao ecossistema terrestre e aos seres humanos: o carvão, a madeira e principalmente o petróleo. Em paralelo, outras fontes renováveis como a energia hidrelétrica surgiram nas décadas iniciais da “indústria” da energia elétrica, porém a nível global nunca ultrapassou percentualmente o conjunto de fontes poluentes. Conforme a ciência evoluiu na metade do século XX, outras fontes surgiram, com destaque a nuclear (que apesar do seu potencial e controvérsias, nunca chegou a ser parte significativa da matriz global), eólica e a solar.

Assim como na contextualização dos parágrafos anteriores, a energia solar segue um caminho natural de desenvolvimento e aprimoramento, aumentando cada vez mais sua participação como matriz energética e tornando-se um player de destaque em todo o mundo. O sol é uma fonte abundante de energia; apesar de ser finita, ao levar em consideração nossos requisitos energéticos a estrela da Via Láctea mais próxima ao planeta Terra é praticamente uma fonte infinita de energia. Dentre as maneiras que nós descobrimos de transformar a energia do sol em energia outro tipo de energia, a elétrica é a principal, através do efeito fotovoltaico, a partir de conjuntos de células solares dispostas em arranjos conhecidos como módulos solares. Ela, inclusive, foi uma das principais soluções que permitiu a exploração espacial, já que a utilização de energia nuclear é muito arriscada por conta do alto risco de explosões em lançamentos e combustível líquido ou sólido possuem uma autonomia muito menor e necessitariam de constantes reabastecimentos.

Junto com a popularização dos módulos fotovoltaicos surge um problema do ponto de vista físico e comercial: a dificuldade de estimar com precisão a energia gerada por conta de imprecisões em dados meteorológicos para estimar a componente de irradiação que um módulo fotovoltaico inclinado e com determinada orientação (azimute) está submetido. Por mais que, como será abordado, há inúmeros modelos e equações bem definidas, de nada adianta utilizá-las com valores imprecisos de irradiação, temperatura e outras variáveis.

Este trabalho tem como objetivo propor melhorias e levantar discussões sobre como otimizar e melhorar a acurácia das simulações através de um código em python que utiliza um algoritmo de regressão não-linear baseado no algoritmo de Levenberg-Marquadt para a otimização de mínimos quadrados, a fim de tratar os outliers – valores muito afastados da curva de tendência – para que os resultados possuem menor diferença percentual em relação a um valor considerado mais fidedigno. Os valores de referência

foram os da simulação em dois softwares, o PVsyst que é o mais utilizado no mercado e com mais estudos e pesquisas usando-o como referência e o HelioScope, entretanto a intenção é que o código possa ser utilizado com qualquer outra referência, como outros softwares ou medidas em campo através de instrumentos para aferir a potência e consequentemente energia de um módulo fotovoltaico inclinado monofacial. Não serão abordados os casos de módulos bifaciais ou com solar track (seguidores solar) por ser um código piloto e essas nuances solicitarem uma maior complexidade a nível computacional para uma maior eficácia, além de possuírem menos estudos e pesquisas a respeito do que os módulos bifaciais montados em uma estrutura fixa como um telhado, uma laje e até mesmo os módulos em estruturas em solo.

Um ponto importante a se destacar é que, por mais que nas simulações há o fornecimento dos valores de energia que um determinado sistema fotovoltaico providencia em determinadas condições, o escopo do trabalho destina-se somente a comparar com a irradiância do coletor ou do "plane of array", sendo indiferente a configuração de inversores, quantidade de módulos ou outros fatores. Na simulação o que importa são as coordenadas do local (longitude e latitude), inclinação (tilt) do módulo, azimute, os dados meteorológicos e o modelo de transposição, que será destacado a parte mais adiante. A irradiância direta normal e refletida pelo solo possuem formulações semelhantes em ambos os softwares, porém mesmo que fossem totalmente distintas o código proposto ainda assim funcionaria visto que a diferença de simulação entre os softwares, nas mesmas condições, é relativamente pequena ao longo de um ano inteiro.

1.2 Breve história da energia solar

A história do desenvolvimento da energia solar remonta milênio atrás, com os primeiros registros de utilização da energia solar por civilizações antigas como egípcios, os gregos e os romanos, por exemplo. No entanto, foi no século XIX que ocorreram avanços significativos no campo da energia solar.

Em 1839, o físico francês Alexandre Edmond Becquerel descobriu o efeito fotovoltaico, que é a capacidade de certos materiais conduzirem corrente elétrica quando expostos aos fótons, no caso aos fótons provenientes do Sol. Ao longo do século XX, progressos notáveis no desenvolvimento da energia solar foram observados, como em 1954, quando a Bell Laboratories criou a primeira célula solar de silício capaz de converter a luz solar em eletricidade de forma eficiente. Nas décadas seguintes, avanços tecnológicos significativos foram realizados, resultando em células solares cada vez mais eficientes e com custos de produção mais baixos, a partir do domínio do processo de manufatura das mesmas e de extração dos componentes necessários para sua fabricação (dentre eles, o Silício).

Na década de 1970, a crise do petróleo despertou um interesse renovado pela energia solar, com governos e instituições de pesquisa investindo em programas de pesquisa e desenvolvimento para impulsionar a tecnologia solar. A partir deste momento, a energia solar começou a se expandir rapidamente em todo o mundo, impulsionada por políticas de incentivo, avanços tecnológicos e uma crescente conscientização sobre a importância das energias renováveis na matriz energética regional e global.

Atualmente, a energia solar é uma das fontes de energia mais promissoras e de crescimento mais rápido globalmente. A capacidade instalada de energia solar tem aumentado exponencialmente nas últimas décadas, impulsionada por investimentos maciços, redução de custos e avanços tecnológicos contínuos, desempenhando um papel fundamental na transição global para uma matriz energética mais limpa e sustentável.

1.2.1 Energia solar no Brasil

A história da energia solar no Brasil começa na década de 1970, durante a crise do petróleo, quando o país começou a explorar a energia solar como uma alternativa aos combustíveis fósseis que estavam com uma grande valorização por conta de sua escassez. O Brasil, com sua localização geográfica favorável e abundância de luz solar em detrimento de outras regiões de altas latitudes e que possuem baixos índices de irradiação, logo percebeu o potencial da energia solar.

No início, a energia solar era usada principalmente para aquecimento de água em residências e empresas. Com o desenvolvimento da tecnologia e redução do custo, o governo brasileiro e algumas empresas começaram a explorar a energia solar fotovoltaica comercialmente. Na década de 1990, o Brasil começou a implementar políticas para promover a energia solar, isso incluiu a criação de incentivos fiscais para a

Projeto de Graduação

instalação de painéis solares. Em 2004, o Brasil lançou o Programa de Incentivo à Energia Solar (PROINFA), que visava aumentar a capacidade de energia solar do país. O programa foi um sucesso, resultando em um aumento significativo na instalação de painéis solares em todo o país. Apesar desses avanços, a energia solar ainda representava uma pequena porcentagem da matriz energética do Brasil.

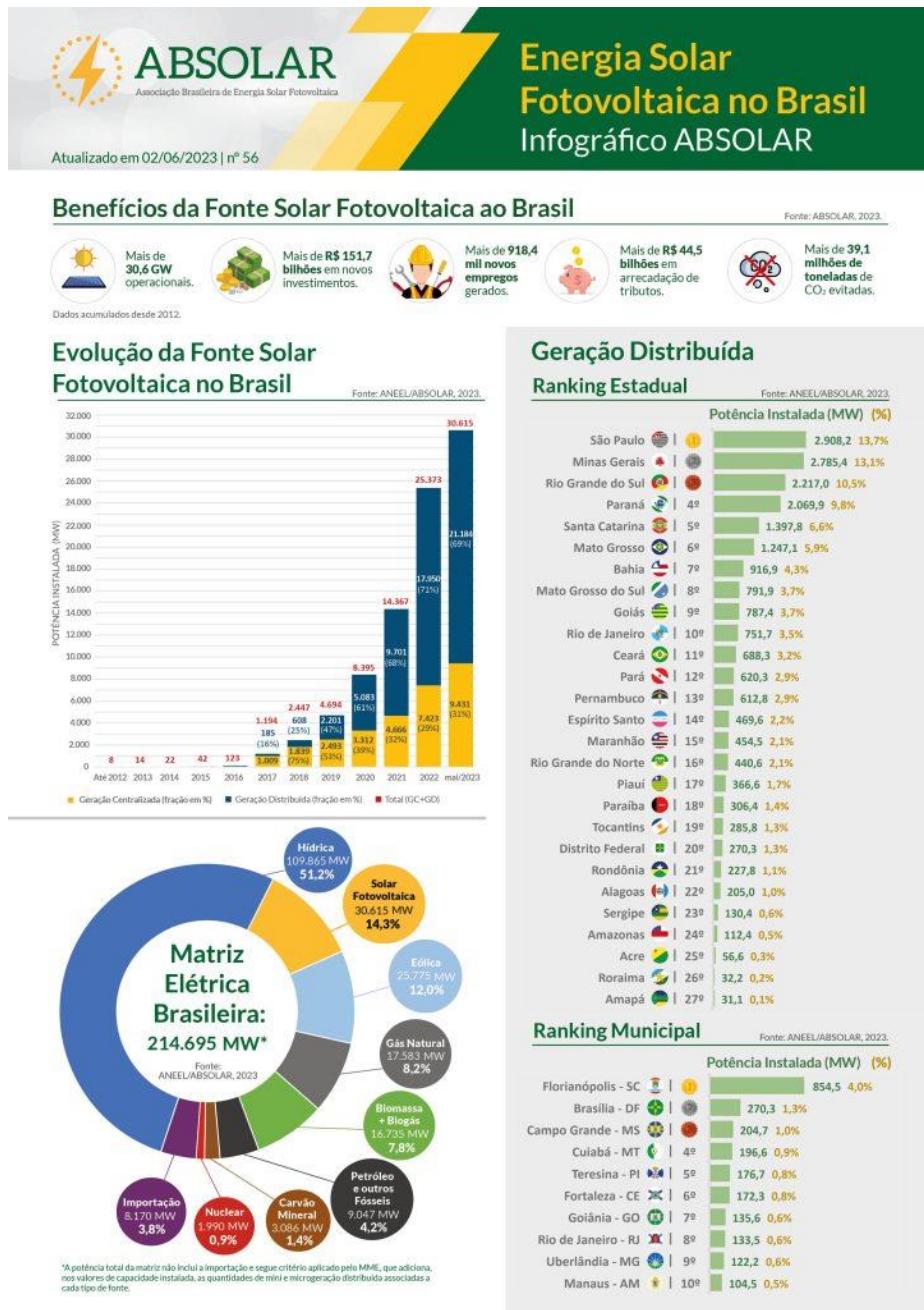


Figura 1- Infográfico 1: Panorama da solar fotovoltaica no Brasil e no mundo (ABSOLAR/ANEEL, 2023)

Foi no ano de 2012 que a ANEEL possibilitou o uso da Geração Distribuída (GD) como alternativa para pessoas físicas e jurídicas (como empresas) instalarem sistemas de energia solar conectados à rede (on-grid) e compensassem o excedente com créditos de energia, ampliando de forma significativa de vez a capacidade instalada no Brasil. Desde 2020 a GD ultrapassou a capacidade instalada da Geração Centralizada, com ambas, em 2023, ultrapassando 30 GW e 2 milhões de sistemas conectados à rede, tornando-se a segunda fonte na matriz elétrica brasileira, somente atrás da hidráulica.

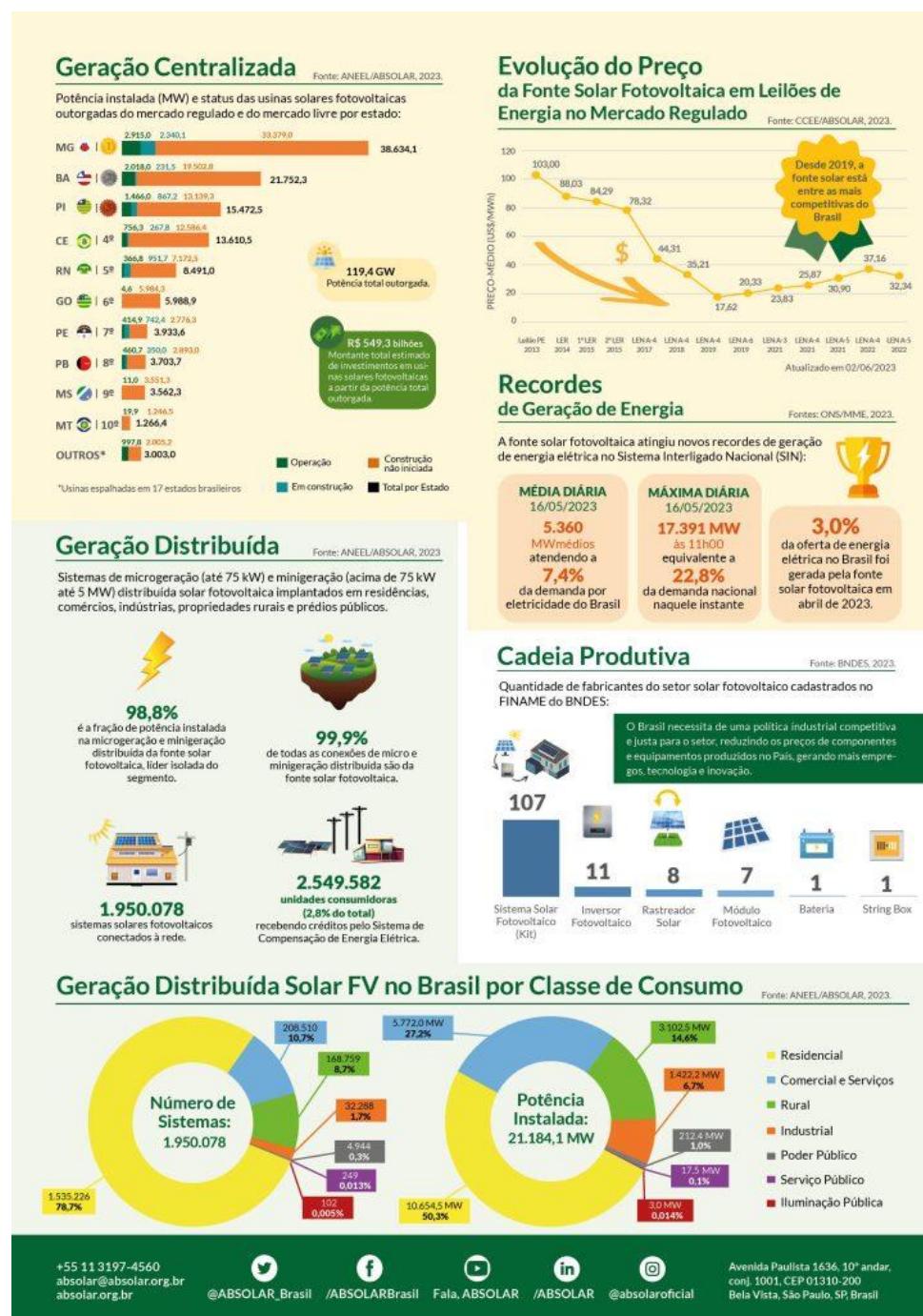


Figura 2 - Infográfico 2: Panorama da solar fotovoltaica no Brasil e no mundo (ABSOLAR/ANEEL, 2023)

Em 2012 também ocorreu o primeiro leilão de energia solar no Brasil, no qual empresas de energia solar competiram para fornecer eletricidade à rede. O leilão foi realizado com sucesso, resultando em contratos para a construção de várias usinas solares em todo o país. Desde então, o Brasil tem realizado leilões de energia solar regularmente, resultando em um rápido crescimento da capacidade de energia solar do país e de acordo com a ABSOLAR, desde 2019 a energia solar está entre as fontes mais competitivas do Brasil, com 119,4 GW de potência com outorgada (capacidade total autorizada, não necessariamente a potência em implantação, com grande percentual com construção não iniciada de acordo com o infográfico da ABSOLAR). Os infográficos da ABSOLAR citados acima possuem informações retiradas da ANEEL até junho de 2023.

1.3 Irradiância no coletor ou no plano (plane of array)

A Irradiância no coletor ou no plano de array, também conhecida como Plane of Array (POA), é um conceito fundamental para a energia solar. Ela é a medida da intensidade da energia solar que incide em uma superfície coletora, seja um painel solar fotovoltaico ou um coletor solar térmico (placa de aquecimento solar), e sua potência é usualmente medida em watts por metro quadrado (W/m^2), equivalente a irradiância solar sobre ela. A energia solar que alcança a superfície da Terra é composta por radiação direta (horizontal e normal) e difusa, sendo que a radiação direta vem diretamente do sol, enquanto a radiação difusa é a luz solar que foi refratada e espalhada pela atmosfera. Ambas contribuem para a irradiância no coletor ou no plano de array, com uma terceira contribuição dependendo da ocasião, da irradiância refletida pelas superfícies ao redor.

O valor da irradiância no plano é influenciado por vários fatores: Primeiro, a posição do sol no céu, que muda ao longo do dia e do ano, definida como o ângulo solar e é um fator crítico no cálculo da irradiância. Segundo, as condições atmosféricas, como a quantidade de nuvens ou a presença de partículas na atmosfera, que podem diminuir a quantidade de radiação solar que chega ao coletor ao longo do dia. Outro fator, por exemplo, é a orientação e a inclinação do coletor.

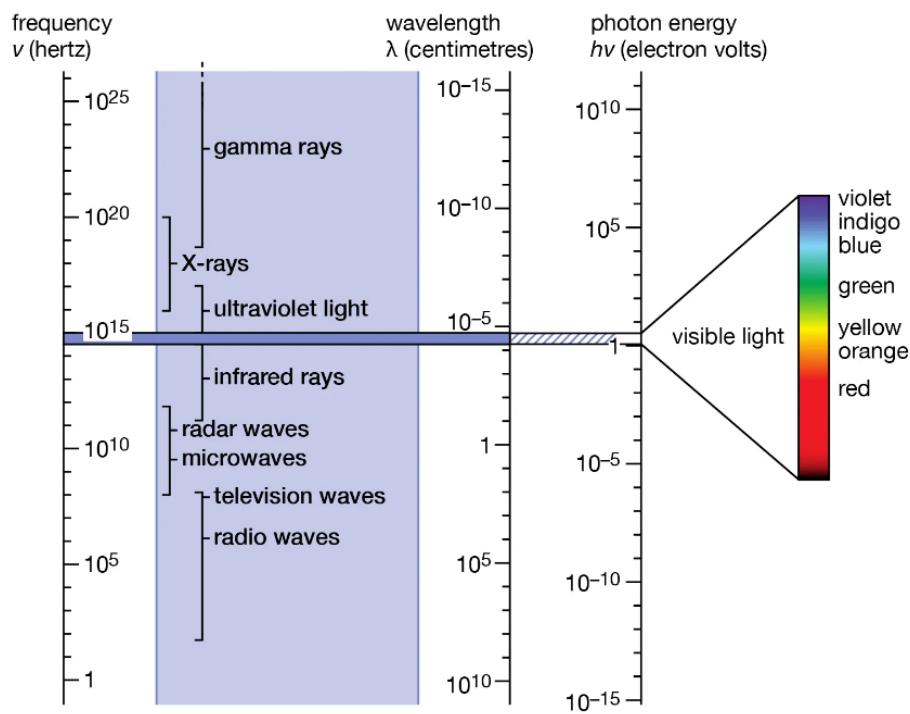
O cálculo da irradiância no coletor é uma parte crucial do projeto de um sistema de energia solar, pois determina a quantidade de energia que o sistema pode gerar. Cálculos precisos podem ajudar a otimizar o desempenho do sistema e garantir que ele seja capaz de atender às demandas de energia. Além disso, o entendimento dessas medidas também é fundamental para os esforços de manutenção e monitoramento do sistema de energia solar. A análise da irradiância no coletor é uma área ativa de pesquisa, com cientistas constantemente desenvolvendo novas maneiras de melhorar a eficiência dos coletores solares. A compreensão e a otimização desta medida podem levar a avanços significativos na eficiência e na eficácia dos sistemas de energia solar, tornando-os uma opção de energia ainda mais atraente, por isso o objetivo do trabalho é melhorar a compreensão através do tratamento dos pontos fora da(s) curva(s) de tendência(s) da irradiação em determinado local, com um ângulo e azimute definido previamente.

2. Ondas eletromagnéticas e fótons

As ondas eletromagnéticas são uma forma fundamental de energia que se move através do espaço (vácuo) a uma velocidade constante, conhecida como a velocidade da luz. Essas ondas são geradas por uma variedade de fontes, incluindo átomos excitados, oscilações de partículas carregadas e até mesmo certos tipos de partículas subatômicas.

As ondas eletromagnéticas são caracterizadas basicamente por sua frequência ou o comprimento de onda que possui uma relação inversa à frequência, que é o número de oscilações por segundo, e sua amplitude, que é a altura da onda. O espectro de freqüências de radiação eletromagnética se estende desde valores muito altos de comprimento de onda do espectro infravermelho (alguns metros para rádio FM e centenas de metros para rádio AM), passando pela pequena faixa de luz visível e além, para o espectro ultravioleta, como raios-X e raios gama de valores substancialmente mais altos de frequência e comprimento de onda infinitesimais.

As ondas eletromagnéticos podem ser classificadas também pela energia de fóton, como ilustrado na figura abaixo.



© Encyclopædia Britannica, Inc.

Figura 3 - Faixa espectral de radiação (Encyclopaedia Britannica, 2023)

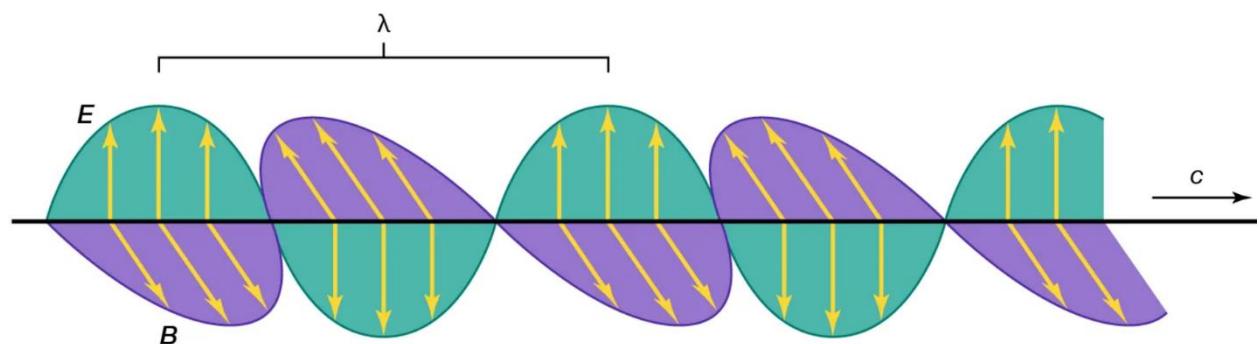
Os fótons são partículas elementares que representam quantidades quânticas de luz e outras formas de radiação eletromagnética. O conceito de fóton foi introduzido por Albert Einstein em 1905 para explicar o efeito fotoelétrico, no qual ele propôs a existência de pacotes discretos de energia durante a transmissão de luz. O termo "fóton" (do grego "phōs, phōtos", que significa "luz") não foi usado até 1926.

Eles são únicos porque não possuem massa e sempre se movem à velocidade da luz no vácuo. A energia de um fóton depende da frequência da radiação eletromagnética; existem fótons de todas as energias, desde raios gama e raios-X de alta energia, passando pela luz visível, até ondas de rádio e infravermelhas de baixa energia. O Sol não emite fótons em todas as freqüências possíveis, como será examinado mais adiante. Outro ponto interessante a levar em consideração é que a distribuição da energia da radiação solar não é uniforme em todo o seu espectro de frequência emitido, por mais que a relação acima da relação proporcional entre aumento de frequência e aumento da energia do fóton ser verdadeira, levando

a conclusão de que o sol emite os fôtons de diferentes frequências em forma desigual, seguindo aproximadamente o comportamento de um corpo negro.

2.1 Básico da Teoria Quântica

A teoria quântica, também conhecida como mecânica quântica, é um dos pilares fundamentais da física moderna, descrevendo o comportamento das partículas subatômicas e de suas interações entre si. A teoria quântica foi desenvolvida no início do século XX por uma série de cientistas, com especial destaque para Max Planck, Albert Einstein, Niels Bohr e Erwin Schrödinger.



© Encyclopædia Britannica, Inc.

Figura 4 - Representação gráfica da onda eletromagnética (Encyclopædia Britannica, 2023)

Uma das principais bases da teoria quântica é o princípio da dualidade onda-partícula, que afirma que as partículas subatômicas podem se comportar tanto como partículas discretas quanto como ondas contínuas. Isso significa que as partículas podem exibir características tanto de partículas localizadas em posições específicas quanto de ondas espalhadas em um espaço contínuo, independentemente do meio.

Outro aspecto fundamental da teoria quântica é o conceito de superposição. De acordo com a teoria, uma partícula pode existir em múltiplos estados simultaneamente, até que seja medida ou observada. A medida de uma partícula em um determinado estado faz com que ela colapse em um estado específico, com o famoso experimento mental do Gato de Schrödinger sendo o exemplo mais citado que descreve a superposição.

A teoria quântica também introduziu o conceito de probabilidade quântica. Em vez de determinismo absoluto, a teoria quântica descreve o comportamento das partículas em termos de probabilidades. As propriedades de uma partícula podem ser expressas por meio de uma função de onda, que contém informações sobre a probabilidade de a partícula ser encontrada em diferentes estados.

A teoria quântica revolucionou a compreensão do mundo subatômico e tem aplicações em diversas áreas da ciência e tecnologia, incluindo a física de materiais, a eletrônica quântica, a computação quântica e o desenvolvimento da energia fotovoltaica.

2.2 Espectro eletromagnético do Sol

O espectro eletromagnético do Sol abrange uma ampla faixa de comprimentos de onda, desde raios gama de alta energia até ondas de rádio de baixa energia. Esse espectro é uma representação das diferentes formas de radiação eletromagnética emitidas pelo Sol, que é uma estrela de tipo G com uma temperatura superficial de cerca de 5.500 graus Celsius. A radiação solar segue uma distribuição aproximada de corpo negro, o que significa que a intensidade da radiação em cada comprimento de onda está relacionada à temperatura do corpo emissor.

De acordo com a lei do deslocamento de Wien, à medida que a temperatura aumenta, o pico de emissão se desloca para comprimentos de onda menores. No caso do Sol, o pico de emissão ocorre na faixa do

espectro visível, mais especificamente na região do amarelo-esverdeado. Essa é a razão pela qual vemos o Sol como uma fonte de luz branca quando observado a olho nu, desconsiderado as distorções atmosféricas que ocorrem por conta de gases ou por conta da curvatura terrestre (quando o sol nasce e ao se pôr no horizonte).

Além do espectro visível, o Sol emite radiação em outras regiões do espectro eletromagnético. A radiação ultravioleta (UV) tem comprimentos de onda mais curtos do que a luz visível e é dividida em três faixas: UV-A, UV-B e UV-C. A radiação UV (em sua maior parte a UV-C e UV-B) é absorvida pela camada de ozônio na atmosfera terrestre, por não conseguir absorver perfeitamente toda faixa UV há o risco de câncer de pele e outras doenças uma exposição excessiva da pele ao Sol aos fótons de alta energia. A radiação infravermelha (IR) tem comprimentos de onda mais longos do que a luz visível e é percebida como calor, apesar de possuírem uma menor energia do que os fótons no espectro UV, como demonstrado por William Herschel.

A compreensão do espectro eletromagnético do Sol é crucial para a ciência solar e para o desenvolvimento de tecnologias solares. Os painéis solares fotovoltaicos são projetados para converter a luz solar em eletricidade, aproveitando a radiação na faixa do espectro visível.

2.3 Constante Solar

A constante solar (G_{sc}) é uma unidade de medida equivalente a densidade de fluxo do sol a uma distância de 1 unidade astronômica (au, aproximadamente 150 milhões de quilômetros). É um parâmetro fundamental nas áreas de energia solar e ciências atmosféricas. Ela representa a quantidade de radiação eletromagnética solar (radiação extraterrestre) recebida por unidade de área fora da atmosfera da Terra, em uma superfície normal aos raios solares. O valor amplamente aceito para a constante solar é de aproximadamente 1361 Watts por metro quadrado (W/m^2), apesar de ligeiras variações superiores ou inferiores a esse valor ao longo dos anos.

A constante solar é um valor de referência essencial usado em diversas aplicações, incluindo cálculos de energia solar, modelagem climática e pesquisas atmosféricas. Seu valor serve como uma base para estimar a energia fotovoltaica e auxilia na avaliação do potencial para geração de energia solar através dos modelos de transposição e de temperatura das células fotovoltaicas. Não é exatamente $1361 W/m^2$ porque pode variar ligeiramente devido a fatores como a atividade solar e seus ciclos, variações na distância Terra-Sol devido a sua órbita elíptica (apesar de possuir uma excentricidade muito pequena e próximo a 0) e incertezas nas medições. No entanto, em períodos curtos, essas variações são relativamente pequenas e não afetam significativamente as suas aplicações.

Contudo é importante destacar que a constante solar representa a irradiância solar total integrada em todas as faixas de comprimento de onda que o Sol emite, incluindo a radiação ultravioleta, a visível e a infravermelha, sendo que para aplicações como as do módulo fotovoltaico de silício (ou outros materiais) nem todo o espectro é aproveitado. Esse valor fornece uma referência para a energia total recebida do Sol e serve como um padrão para o desempenho de sistemas de energia solar e estimativas de produção de energia.

3. Efeitos advindos das ondas eletromagnéticas

O efeito fotovoltaico é o efeito responsável pela utilização da energia solar como uma fonte limpa e renovável. Ele permite a geração de eletricidade a partir da luz solar, porém é muito confundido com outro efeito, o fotoelétrico, que possibilita a produção de corrente elétrica a partir da iluminação de superfícies metálicas. Apesar de outros efeitos como a fotossíntese serem decorrentes da irradiação solar, é importante destacar o efeito fotoelétrico que foi o precursor – por assim dizer – para uma mudança da teoria clássica do eletromagnetismo e consequentemente a descoberta de outros efeitos como o fotovoltaico.

3.1 Efeito fotoelétrico

O efeito fotoelétrico é o fenômeno pelo qual elétrons são liberados de uma superfície quando essa superfície é iluminada por luz. Esse efeito foi observado pela primeira vez por Heinrich Hertz em 1887 quando investigava a natureza eletromagnética da luz e posteriormente explicado por Albert Einstein em 1905, pela teoria dos fótons (a qual inclusive lhe rendeu o Nobel de Física, erroneamente associado aos seus estudos de relatividade geral).

De acordo com suas observações, a luz, composta por fótons que possuem energia proporcional à sua frequência, quando incide em uma superfície metálica pode transferir sua energia para um elétron, liberando-o da superfície. Hertz constatou que o fenômeno não era de natureza eletrostática, porque não havia diferença se a proteção contra a dispersão da luz ao redor do seu experimento fosse condutora ou não. Uma corrente elétrica pode surgir em um circuito a partir do efeito fotoelétrico, como demonstrado na figura abaixo que demonstra o experimento de Thomson para provar que o efeito fotoelétrico ocasionava a emissão de elétrons.

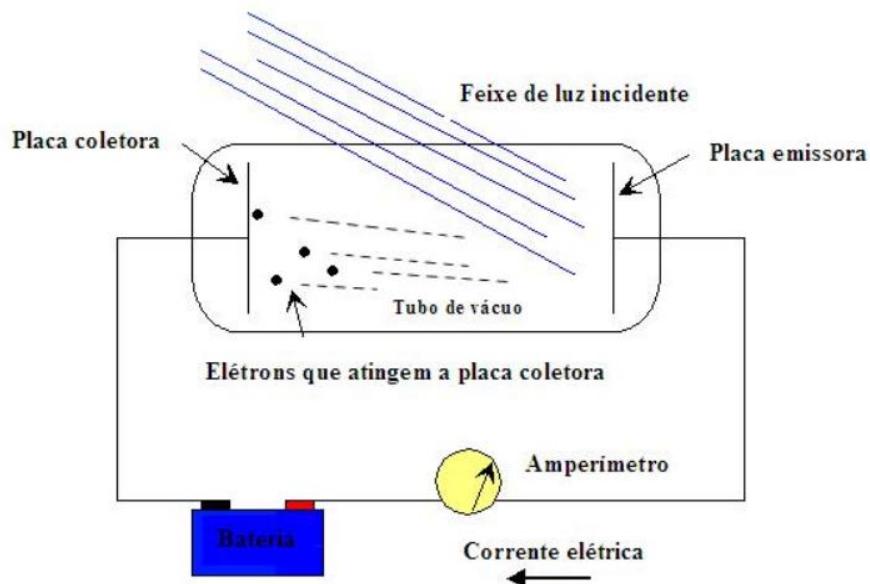


Figura 5 – Experimento do Efeito Fotoelétrico (UFRGS, s.d.)

Depois que Leonard demonstrou em 1903 que a energia dos elétrons emitida não era proporcional a intensidade de luz e que Schweidler demonstrou que era proporcional à frequência ao invés da intensidade de Luz surgiram questionamentos a teoria clássica do eletromagnetismo. Com isso Einstein, a partir da proposta de Planck em 1900, conseguiu explicar o efeito fotoelétrico. Depois que Leonard demonstrou em 1903 que a energia dos elétrons emitida não era proporcional a intensidade de luz e que Schweidler demonstrou que era proporcional à frequência ao invés da intensidade de Luz surgiram questionamentos a teoria clássica do eletromagnetismo. Com isso Einstein, a partir da proposta de Planck

em 1900 sobre a radiação de corpo negro onde os osciladores só poderiam emitir energias em quantidades inteiras de $h\nu$ (sendo h a constante de Planck e ν a frequência de radiação), conseguiu explicar o efeito fotoelétrico e deu início a teoria quântica.

A partir da ideia inovadora de Planck foi proposto que a luz, ao invés de ser considerada como onda, fosse considerada como corpúsculos (fótons) que transmitiam energia, inclusive uma ideia defendida séculos antes por Newton. Einstein propôs que a Energia do elétron ejetado fosse proporcional a $h\nu$ (sendo v a velocidade da luz incidente) menos a função de trabalho do metal, portanto dependendo dessa relação poderia não existir corrente elétrica no arranjo experimental mencionado acima.

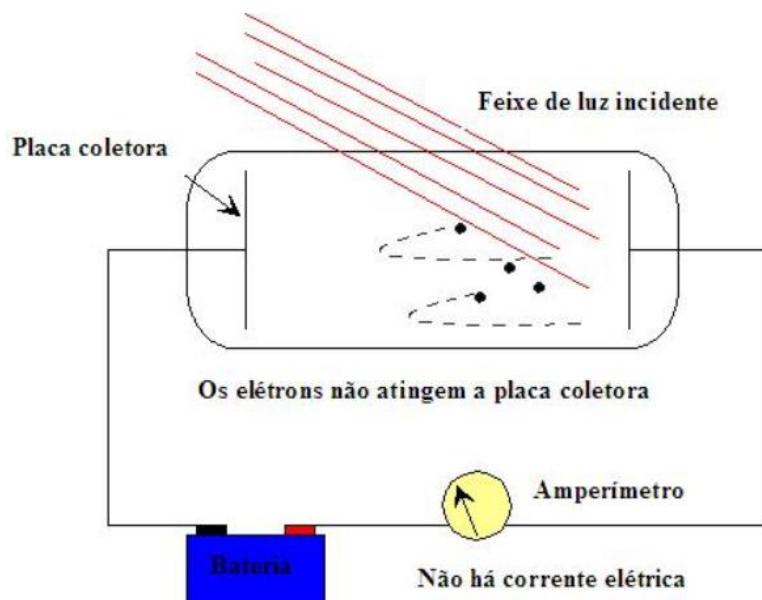


Figura 6 - Relação energia e trabalho do metal (UFRGS, s.d.)

3.2 Efeito fotovoltaico

A base da tecnologia dos módulos fotovoltaicos é o efeito fotovoltaico, proporcionando a conversão direta da energia dos fótons do Sol em corrente contínua, mais tarde usualmente convertida em corrente alternada. Esse efeito ocorre quando a luz incide em um material semicondutor, nos módulos a nível comercial compostos atualmente em sua maior parte por células de silício monocristalino, causando a excitação de elétrons, gerando uma diferença de potencial elétrico. Essa diferença de potencial cria um fluxo de elétrons nos pares buraco-elétron das junções PN das células entre materiais do tipo "n" e materiais do tipo "p" (n-type e p-type), gerando uma corrente elétrica contínua. Um arranjo de dezenas ou centenas das células em Wafers e dos Wafers em módulos fotovoltaicos possibilita a utilização para a produção de módulos de centenas de Wp (Watt-pico).

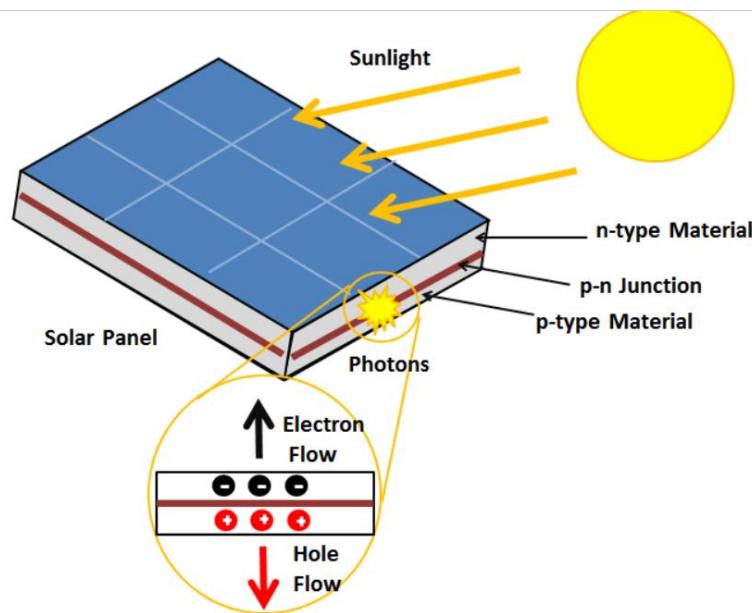


Figura 7 – Efeito Fotovoltaico (Energy Education (University of Calgary), s.d.)

4. Módulos fotovoltaicos e energia solar

Os módulos fotovoltaicos, também conhecidos como painéis solares, são dispositivos que convertem a luz solar diretamente em eletricidade por meio do efeito fotovoltaico. Eles são compostos por células solares, geralmente feitas de silício, que possuem propriedades semicondutoras. Quando a luz solar incide sobre as células solares, os fótons da luz transferem sua energia para elétrons, criando um fluxo de corrente elétrica.

Os módulos fotovoltaicos têm evoluído significativamente ao longo das últimas décadas em termos de eficiência, durabilidade e custo. A eficiência dos módulos fotovoltaicos refere-se à capacidade de converter a luz solar em eletricidade, sendo expressa como a proporção da energia da luz solar incidente que é convertida em energia elétrica de fato. Atualmente, os módulos fotovoltaicos comerciais têm eficiências que variam de cerca de 20%, dependendo do tipo de tecnologia utilizada. A durabilidade dos módulos fotovoltaicos é importante para garantir que eles possam resistir a condições climáticas adversas e permanecer operacionais por um longo período de tempo, com os fabricantes usualmente garantindo uma degradação linear de potência por 12 anos e de garantia de módulos de 25 a 30 anos devido a alta confiabilidade de sua produção.

As tecnologias de módulos fotovoltaicos mais comuns são baseadas em silício cristalino, que pode ser dividido em duas categorias principais: silício monocristalino e silício policristalino, sendo o primeiro o mais utilizado e eficiente atualmente. Além disso, existem tecnologias avançadas em desenvolvimento, como os módulos de película fina, que são feitos de materiais semicondutores diferentes do silício, como o telureto de cádmio (CdTe) e o disseleneto de cobre e índio (CIGS).

4.1 Concepção do módulo fotovoltaico

Os módulos fotovoltaicos são projetados com base em uma série de elementos e concepções para garantir sua eficiência e durabilidade. A concepção dos módulos fotovoltaicos envolve a seleção dos materiais adequados, o design da estrutura e a configuração das células solares. O principal componente dos módulos fotovoltaicos é a célula solar, que é responsável por converter a luz solar em eletricidade. As células solares são geralmente feitas de silício e são projetadas para criar uma camada de campo elétrico, onde os elétrons são liberados quando a luz solar incide sobre elas.

A estrutura dos módulos fotovoltaicos é projetada para proteger as células solares contra danos físicos e ambientais. Geralmente, os módulos são encapsulados em camadas de materiais como vidro temperado na frente e plástico resistente na parte de trás, quando não são bifaciais (que possuem uma camada de vidro atrás também). Essas camadas ajudam a proteger as células solares contra a umidade, poeira, impactos e alguns espectros dos raios ultravioleta.

Além disso, a configuração das células solares nos módulos fotovoltaicos é importante para otimizar a geração de eletricidade. Existem diferentes arranjos de células (séries e paralelo) que influenciam a tensão e a corrente geradas pelos módulos. O design da configuração das células leva em consideração fatores como a resistência interna, perdas de sombreamento e o balanceamento da tensão.

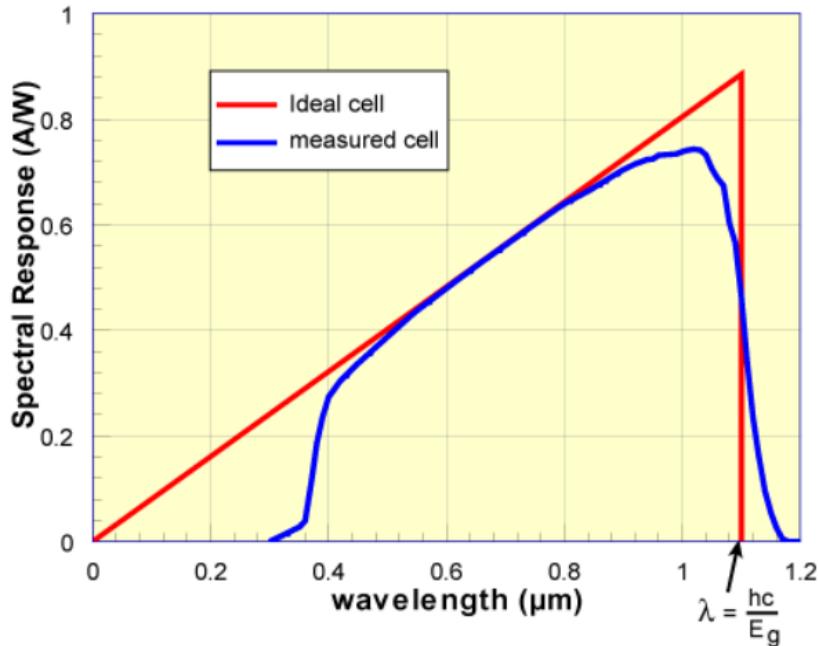


Figura 8 – Resposta espectral da célula de silício (PVEducation, s.d.)

Para ondas abaixo de 400nm o vidro absorve a maior parte da luz e a resposta da célula é muito baixa, enquanto em comprimentos intermediários de onda a resposta é próxima da ideal. Em comprimentos de ondas muito longos a resposta volta a ser nula.

4.2 Modelo de 1 diodo e suas variações

O modelo de 1 diodo é uma representação equivalente em circuito que define a curva IV do módulo fotovoltaico como uma função contínua para certos parâmetros como a corrente de saturação inversa e o fator de qualidade, são obtidos por meio de testes experimentais ou estimados com base em dados fornecidos pelo fabricante do módulo. Nesse modelo, considera-se que o módulo é equivalente a uma célula solar com um único diodo em paralelo. Ele é amplamente utilizado devido à sua simplicidade e facilidade de implementação, devido aos seus princípios físicos (GRAY, 2011).

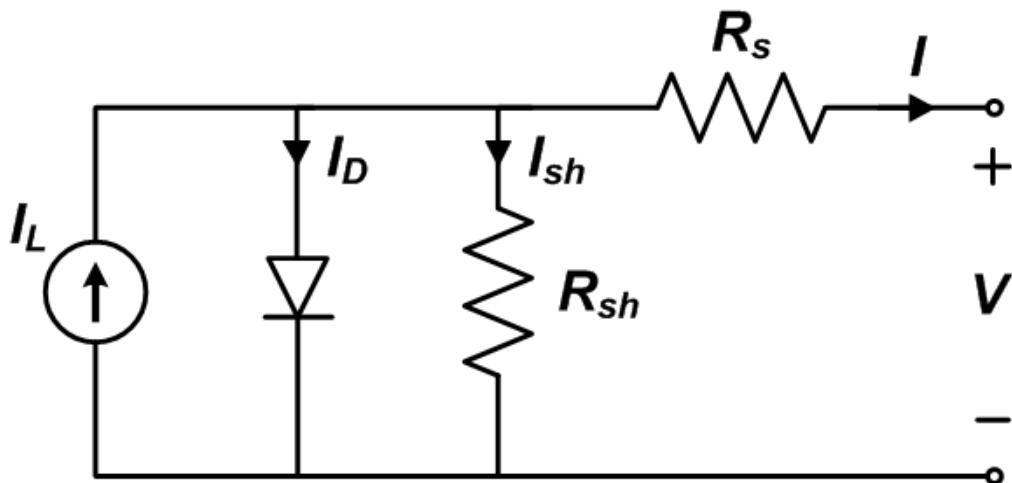


Figura 9 - Modelo de 1 diodo (PVPMC SANDIA, s.d.)

No modelo de 1 diodo, a corrente do diodo é calculada com base na tensão aplicada ao módulo, a intensidade de luz incidente e os parâmetros do diodo, como a corrente de saturação inversa e o fator de qualidade. A corrente gerada pelo módulo é então calculada subtraíndo-se a corrente do diodo da corrente de curto-circuito. A equação que flui pelo diodo I_D pode ser modelada pela equação de Shockley abaixo, considerando I_0 com a corrente de saturação reversa do diodo, n como o fator de idealidade do diodo, R_s a resistência série e V_t como a tensão térmica $\frac{kT_c}{q}$:

$$I_D = I_0 * \left(\exp \left[\frac{(V + I R_s)}{n V_t} \right] - 1 \right)$$

Considerando N_s como o número de células em série e o índice M indicando a tensão e corrente do módulo proposto por (TIAN, 2012):

$$I_M = I_{sc} - I_0 * \left(\exp \left[\frac{(V + I_M N_s R_s)}{n N_s V_t} \right] - 1 \right) - \frac{(V + I_M N_s R_s)}{R_{sh}}$$

Existem variações do modelo de 1 diodo que levam em consideração efeitos adicionais, como a resistência série interna e a resistência paralela de fuga. Essas variações permitem uma melhor representação do comportamento real do módulo fotovoltaico, levando em conta perdas adicionais devido a esses efeitos, como (DE SOTO et al., 2006) e especialmente o modelo do PVsyst, que considera as perdas por corrente de recombinação do módulo.

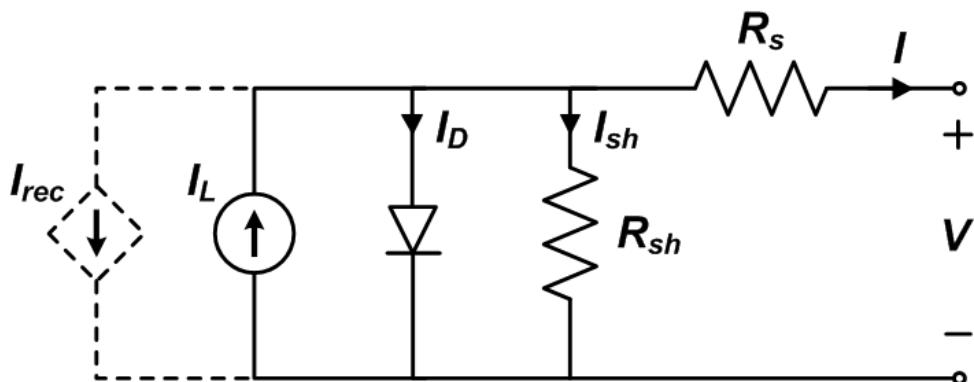


Figura 10 - Modelo de 1 diodo do PVsyst (PVPMC SANDIA, s.d.)

4.3 Classificação quanto aos testes em STC

Os módulos fotovoltaicos passam por testes em condições padrão de teste (STC) para avaliar seu desempenho e características elétricas. Esses testes são conduzidos em laboratórios especializados e seguem normas internacionais para garantir a comparabilidade entre diferentes módulos de diferentes fabricantes. Existem três principais classificações quanto aos testes em STC: potência de pico, corrente de curto-circuito e tensão de circuito aberto, além de outras classificações como a tensão e corrente em máxima potência, eficiência e coeficientes (térmicos) de potência, tensão de circuito aberto e corrente de curto-circuito.

A potência de pico (P_{max}) é a medida da potência máxima que um módulo fotovoltaico pode produzir sob condições ideais de radiação solar e temperatura. Ela é determinada multiplicando-se a tensão de circuito aberto (V_{oc}) pela corrente de curto-circuito (I_{sc}). Essa medida é utilizada para comparar a capacidade de geração de energia entre diferentes módulos, sendo a característica mais destacada dos módulos fotovoltaicos (classificando-os quanto o Wp primeiramente).

A corrente de curto-círcuito (I_{sc}) é a corrente máxima que flui pelo módulo quando os terminais estão em curto-círcuito. Ela é medida em condições de alta irradiação solar. A corrente de curto-círcuito é um parâmetro importante para o dimensionamento do sistema fotovoltaico e para determinar a capacidade de corrente suportada pelo módulo e que o inversor, cabos e demais componentes podem ser submetidos, levando em conta as condições padrão de teste.

A tensão de circuito aberto (V_{oc}) é a tensão máxima que o módulo pode fornecer quando não há carga conectada aos terminais. Ela é medida sob condições de alta irradiação solar. A tensão de circuito aberto é utilizada para determinar a tensão máxima suportada pelo módulo e pode ser utilizada no dimensionamento do sistema fotovoltaico, especialmente em arranjos série onde a tensão dos n módulos do arranjo é somada.

Essas classificações em STC são importantes para padronizar a avaliação dos módulos fotovoltaicos e facilitar a seleção e comparação de diferentes modelos no mercado. No entanto, o desempenho real dos módulos pode variar devido a fatores como a temperatura do ambiente e da célula (a potência decresce de acordo com o aumento de temperatura), sujeira e massa de ar.

4.3.1 Irradiância

A irradiação é a quantidade de energia solar incidente sobre o módulo fotovoltaico por unidade de área. A irradiação é influenciada pela localização geográfica, condições climáticas e ângulo de incidência solar. Quanto maior a irradiação, maior a potência de saída do módulo. Em condições de teste ela é de aproximadamente 1000 W/m^2 , equivalente a um dia muito ensolarado, não necessariamente de Verão, na maior parte da Terra.

4.3.2 Temperatura

A temperatura do módulo fotovoltaico é um fator crítico que afeta seu desempenho. A temperatura de operação do módulo pode ser diferente da temperatura ambiente devido ao aquecimento causado pela radiação solar e a dissipação de calor limitada. A potência de saída do módulo diminui à medida que a temperatura aumenta, seguindo uma curva característica. Em condições de teste a temperatura do módulo é de 25°C . Nota-se que é uma temperatura irreal na maior parte do planeta pois esse não é o valor da temperatura ambiente, sendo que é tipicamente 20°C abaixo da temperatura do módulo.

4.3.3 Massa de ar

A massa de ar é um conceito astronômico definido como a quantidade de ar ao longo do campo de visão de observação de uma estrela ou outro corpo celestional a partir da atmosfera terrestre. Considerando a massa de ar relativa, por definição $AM = 1$ no zênite e $AM = 0$ acima da atmosfera. Conforme o ângulo entre o emissor (neste caso, o Sol) e o zênite aumenta, a massa de ar relativa aumenta, pois é inversamente proporcional ao cosseno, simplificando ao desconsiderar a curvatura terrestre em uma pequena área. É utilizado o espectro solar padrão ASTM G173-03 (ISO 9845-1, 1992) para os testes STC, que considera $AM=1.5$. Valores maiores de massa de ar representam uma curva da irradiação espectral menor, consequentemente um menor valor de irradiação sobre determinada superfície.

4.3.4 Outros parâmetros

Para a classificação em PTC (Photovoltaics for Utility Scale Applications Test Conditions, PVUSA test conditions) da NREL, que considera parâmetros mais realistas, também é considerando uma velocidade média do vento de 1 m/s e uma altura em relação ao nível do mar de 10m, porque foi desenvolvido principalmente para ser utilizado na Califórnia, que possui a maior parte da área urbana próxima ao nível do mar.

5. Componentes da irradiação

A irradiação solar é composta por diferentes componentes que representam a quantidade de energia radiante proveniente do sol e que atinge uma determinada superfície. Essas componentes incluem a Irradiação Global Horizontal (GHI), a Irradiação Direta Normal (DNI), a Irradiação Difusa Horizontal (DHI).

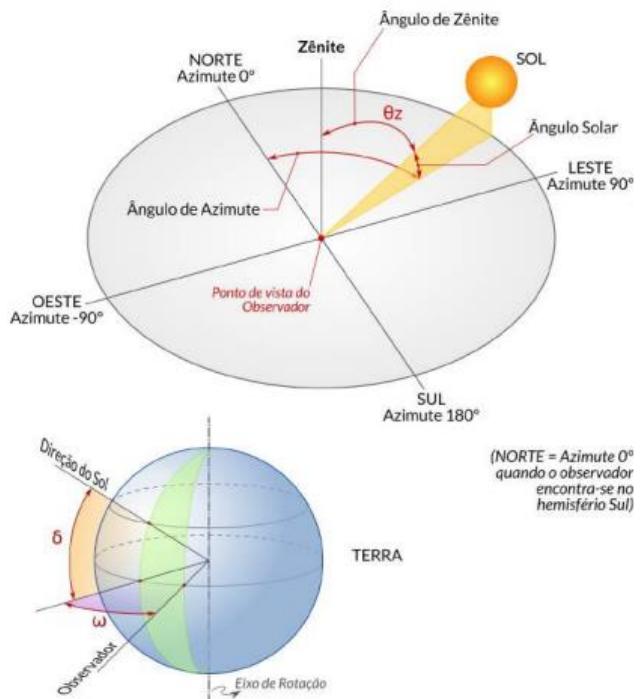


Figura 11 – Ângulos e definições geométricas (Atlas brasileiro de energia solar 2º Ed, 2017)

5.1 GHI

A Irradiação Global Horizontal (GHI) é a quantidade total de radiação solar incidente em uma superfície horizontal, incluindo a radiação direta do sol, a radiação difusa do céu e a radiação refletida do solo. Ela representa a energia total disponível para ser convertida em eletricidade por um sistema fotovoltaico horizontal, o que na prática não acontece (módulos paralelos ao solo) quando a intenção é otimizar a geração de energia de um certo sistema, porém é o principal parâmetro para simulação de geração por conta da consideração de 3 componentes diferentes. A GHI pode ser calculada a partir da DNI, DHI e do ângulo zenital do sol (θ_z), que é o ângulo entre a direção do sol e a vertical. Geralmente ela é calculada negligenciando a componente refletida pelo solo pela incerteza e geralmente a pouca participação dela no valor total. Esta equação assume que não há obstruções ou sombras que possam bloquear a radiação solar direta e que a superfície é plana e horizontal.

$$GHI = DNI \cos(\theta_z) + DHI$$

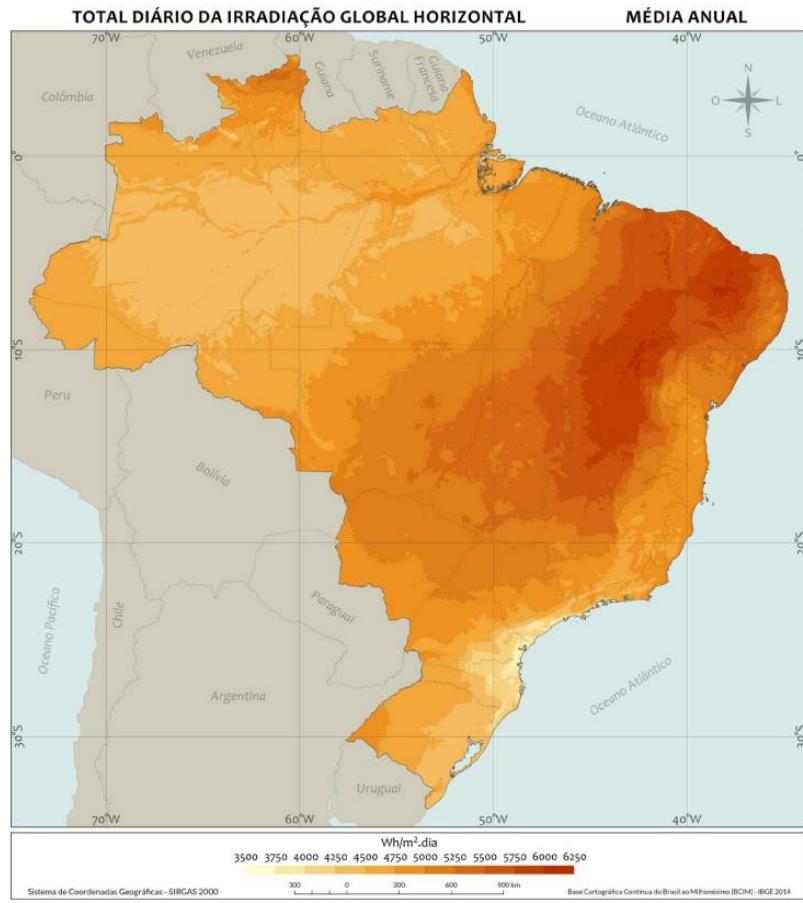


Figura 12 – Mapa da GHI no Brasil (Atlas brasileiro de energia solar 2º Ed, 2017)

Para fins ilustrativos, no Brasil o GHI é maior no Nordeste e no norte de Minas Gerais, no Sul o índice é menor por conta especialmente da alta latitude e na região da Floresta Amazônica o clima diferente (que resulta em mais perdas por partículas d'água por exemplo) explica o motivo de não possuir um GHI tão alto quanto o do Nordeste, mesmo em latitudes semelhantes próximas à Linha do Equador. O mapa está no sistema de coordenadas geográficas SIRGAS 2000, comumente adotado para o Brasil, e utiliza a Base Cartográfica Contínua do Brasil ao Milionésimo (BCIM) disponibilizada pelo IBGE (2014).

5.2 DNI

A irradiação direta normal (DNI) é um parâmetro fundamental na avaliação do potencial solar de uma localização. É definida como a quantidade de radiação solar em uma superfície perpendicular (ou normal) à direção do sol, sem considerar a radiação difusa ou refletida. A DNI é expressa em watts por metro quadrado (W/m^2) e é uma medida da intensidade da radiação solar. A DNI é influenciada por vários fatores, incluindo a posição do sol no céu (determinada pela latitude, dia do ano e hora do dia), a atmosfera e as condições climáticas. Em um dia claro, a DNI pode exceder 1000 W/m^2 ao meio-dia solar. A DNI pode ser calculada a partir da irradiação extraterrestre (I_0) na superfície da Terra, que é a radiação solar recebida fora da atmosfera da Terra, e o índice de claridade (k), que é a razão entre a irradiação solar na superfície da Terra e a irradiação extraterrestre.

$$DNI = I_0 k$$

5.2.1 DNI Extraterrestre

A Irradiância Extraterrestre Direta Normal (DNI Extraterrestre) é uma medida da intensidade da radiação solar fora da atmosfera da Terra, em um plano perpendicular (normal) à direção dos irradiancias solares. Esta medida é importante para a modelagem e simulação de sistemas de energia solar, porque fornece uma referência para a quantidade máxima de energia solar que pode ser recebida na Terra, sem considerar as perdas por reflexão e refração na atmosfera, principalmente por conta da massa de ar que causa atenuação na potência fornecida pelo sol.

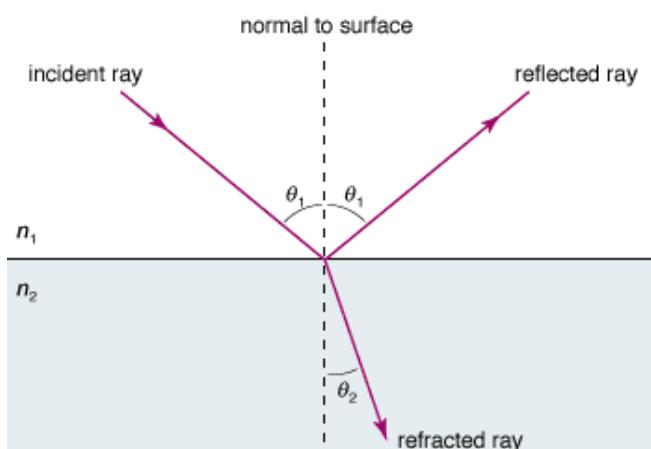
$$I_0 = G_{sc} \left[1 + 0,33 \cos \left((n - 2) \frac{360}{365} \right) \right]$$

5.3 DHI

A irradiação difusa horizontal (DHI) é uma medida da quantidade de radiação solar que atinge a superfície da Terra após ser dispersada pela atmosfera. A dispersão pode ocorrer devido a moléculas de ar, partículas de água nas nuvens, ou partículas de poeira e poluição na atmosfera. A DHI é influenciada por uma variedade de fatores, incluindo a posição do sol no céu, a transparência da atmosfera e as condições climáticas. Em um dia claro, a DHI é tipicamente uma pequena fração da GHI, enquanto em um dia nublado, a DHI pode ser a maior parte da GHI. Devido a Lei de Snell-Descartes é possível perceber que existe uma mudança de direção da irradiação solar que incide sobre a superfície devido a mudança de meio que ocorre, por exemplo, devido a partículas d'água nas nuvens ou gases na atmosfera, ocasionando uma irradiação difusa.

A DHI pode ser calculada a partir da GHI e da irradiação direta normal (DNI) se o ângulo zenital do sol θ_z for conhecido. A relação entre essas quantidades é dada pela equação mencionada para a GHI, apenas adaptando-a:

$$DHI = GHI - DNI \cos(\theta_z)$$



© 2006 Encyclopædia Britannica, Inc.

Figura 13 – Lei da reflexão e refração (Encyclopaedia Britannica, 2023)

5.4 Modelos de Transposição

Os modelos de transposição de irradiação solar são utilizados para estimar as componentes de irradiação, como a Irradiação Global Horizontal (GHI), a Irradiação Direta Normal (DNI), a Irradiação Difusa Horizontal (DHI), a Irradiação na Superfície Inclinada (BHI) e a Irradiação Total Global (GTI). Esses modelos consideram as características da localização geográfica, as condições atmosféricas e os parâmetros do sistema solar, sendo as características físicas através de arquivos meteorológicos usualmente anuais, com valores horários (resultando em cerca de 8700 valores).

A relação entre os modelos de transposição e as componentes de irradiação ocorre por meio de equações e algoritmos que levam em consideração fatores como a latitude, longitude, altitude, ângulos solares, radiação extraterrestre, nebulosidade, entre outros. Esses modelos são baseados em princípios físicos e estatísticos, e são ajustados e validados com dados medidos em estações meteorológicas. Por exemplo, o modelo de transposição de Perez utiliza dados como a GHI e a DHI para calcular a DNI, que é a radiação solar direta incidente em uma superfície inclinada. A partir da DNI, é possível calcular a BHI e a GTI para diferentes ângulos de inclinação e orientações, de acordo com as especificações que os módulos do sistema fotovoltaico estão submetidos.

Outro modelo comumente utilizado é o modelo de transposição de Liu-Jordan, que também estima as componentes de irradiação com base em equações matemáticas e parâmetros atmosféricos. Ele, diferente do Modelo de Perez, considera a influência das nuvens e da nebulosidade na distribuição da radiação solar.

6. Albedo

O albedo é a fração da irradiância horizontal global que é refletida pela superfície. Usualmente o albedo é aproximadamente igual a 0 em uma superfície muito escura e aproximadamente igual a 1 em uma branca ou metálica clara. Não há nenhum corpo com albedo = 0 na natureza, somente um corpo negro possuiria tal propriedade, onde ele absorve toda radiação eletromagnética que incide sobre ele; a luz não o atravessa e nem é refletida. Apesar dessas características, idealmente em equilíbrio termodinâmico ele irradiaria toda energia por ele absorvida, porém não em forma de luz. Da mesma forma, não há nenhum corpo com o albedo = 1 na natureza; tal objeto seria caracterizado como um corpo branco. Um corpo branco seria o oposto do corpo negro, possuindo uma superfície que refletiria toda radiação incidente de forma uniforme em todas as direções.

Em termos de corpo negro e corpo branco é necessário introduzir os conceitos de transmissão, absorção e reflexão para seu entendimento; entretanto o albedo em si é um parâmetro somente da reflexão da irradiância global horizontal, não levando em conta outros tipos de radiação que incidem sobre todos os corpos, portanto não é necessário um maior aprofundamento nos demais conceitos.

6.1 Albedo como constante

O PVsyst, por exemplo, oferece como base as seguintes estimativas de valores para o albedo, inclusive com diferentes cenários para algumas superfícies que mudam de albedo devido a precipitação, sujeira (corrosão do aço) ou como no caso da topografia da grama:

Tabela 1 - Valores do coeficiente de Albedo (Adaptado do PVsyst, s.d.)

Superfície	Albedo mínimo	Albedo constante	Albedo máximo
Ambiente urbano	0,14		0,22
Grama	0,15		0,25
Grama recém-cortada		0,26	
Neve branca		0,82	
Neve molhada	0,55		0,75
Asfalto seco	0,09		0,15
Asfalto molhada		0,18	
Concreto	0,25		0,35
Azulejos vermelhos		0,33	
Alumínio		0,85	
Cobre		0,74	
Aço galvanizado		0,35	
Aço galvanizado sujo		0,08	

6.2 Albedo como variável

A consideração do albedo como uma variável em vez de uma constante é uma abordagem mais precisa ao modelar a irradiação solar. O albedo refere-se à fração da radiação solar refletida por uma superfície, e sua magnitude varia dependendo das características do local, como a presença de vegetação diversificada, neve, água ou superfícies urbanas. Ao considerar o albedo como uma variável, é possível levar em conta sua variação sazonal e espacial, o que pode ter um impacto significativo na quantidade de radiação solar disponível, considerando que os arquivos TMY que geralmente são utilizados pelos modelos de transposição possuírem os dados de albedo.

Vários estudos têm explorado a influência do albedo na irradiação solar, especialmente em regiões com diferentes tipos de cobertura do solo. Por exemplo, áreas urbanas com superfícies de concreto e asfalto tendem a ter um albedo mais baixo, refletindo menos radiação solar em comparação com áreas rurais

com vegetação. A inclusão do albedo como uma variável permite uma modelagem mais precisa da radiação solar incidente em diferentes tipos de superfície. Além disso, a consideração do albedo como uma variável dinâmica também pode ser relevante em regiões onde ocorrem mudanças sazonais significativas, como áreas cobertas de neve no inverno. A cobertura de neve tem um albedo alto, refletindo uma grande quantidade de radiação solar de volta ao espaço. Conforme a neve derrete, o albedo diminui e mais radiação solar é absorvida pela superfície, afetando a disponibilidade de energia solar.

7. Principais modelos de transposição da irradiação

Os modelos de transposição de irradiação solar são ferramentas utilizadas para estimar as componentes de irradiação em diferentes superfícies e inclinações. Esses modelos são fundamentais para o dimensionamento e o planejamento de sistemas solares, permitindo a estimativa da quantidade de radiação solar disponível em determinada localidade. Existem diferentes modelos de transposição que podem ser aplicados de acordo com as necessidades e características do sistema em estudo.

Além dos modelos citados abaixo, existem diversas outras abordagens e algoritmos desenvolvidos para estimar a irradiação solar em diferentes condições atmosféricas e geográficas. Alguns desses modelos são específicos para regiões ou tipos de superfície, como modelos para regiões montanhosas (com baixa pressão atmosférica), áreas urbanas ou superfícies cobertas por neve onde a temperatura e a radiação refletida (devido ao maior albedo) sofrem mudanças sazonais significativas.

É importante, antes, citar a fórmula da irradiação refletida de acordo com condições ideais, considerando $\sum c$ como o ângulo do coletor e α como o albedo da superfície.

$$I_{RC} = I_{GH} \alpha \left(\frac{1 + \cos(\sum c)}{2} \right)$$

7.1 Modelo de Hay-Davies

O Modelo de Hay ou Modelo de Hay-Davies é baseado em uma abordagem simplificada, considerando a radiação direta e difusa em superfícies inclinadas. Ele utiliza fórmulas trigonométricas e dados meteorológicos para estimar as componentes de irradiação solar em diferentes ângulos de inclinação e orientações. Ele depende da constante solar mencionada anteriormente, que indica a irradiância disponível acima da atmosfera terrestre, para que seja possível computar o índice de claridade. Este índice é utilizado posteriormente para calcular a porção circunsolar da irradiância difusa. Na fórmula abaixo n é relativo ao dia do ano.

$$I_0 = G_{sc} \left[1 + 0,33 \cos \left((n - 2) \frac{360}{365} \right) \right]$$

É necessário calcular o índice de claridade K a partir da irradiância direta (beam) horizontal I_{BH} , que é a irradiância global horizontal I_{GH} menos a irradiância difusa horizontal I_{DH} . O coeficiente b é um coeficiente derivado do ângulo solar, sendo $b = \max(0,087, \sin(\beta_s))$ e β_s o ângulo de inclinação solar $\beta_s = 90^\circ - Z_s$ (o ângulo de zênite solar).

$$K = \frac{I_{BH}}{b I_0}$$

Combinando as equações acima é possível descrever a irradiância difusa disponível para um módulo fotovoltaico com inclinação pelo modelo de Hay, considerando $a = \max(0, \cos(\theta))$, sendo θ o ângulo de incidência solar no módulo fotovoltaico:

$$I_{DHC} = I_{DH} \left[(1 - K) \left(\frac{1 + \cos(\sum c)}{2} \right) + K \frac{a}{b} \right]$$

7.2 Modelo de Perez

O modelo de Perez é um dos mais populares e complexos. Ele leva em consideração uma série de parâmetros, como latitude, longitude, altitude, ângulos solares, radiação extraterrestre e nebulosidade. Esse modelo utiliza informações de imagens de satélite e dados de estações meteorológicas para estimar as componentes de irradiação solar em uma superfície inclinada. Abaixo há um fluxograma do Modelo de Perez (elaborado pelos próprios autores) em 1990, indicando os parâmetros de entrada como a temperatura do ponto de orvalho, que é a temperatura de condensação da água, a irradiância direta, a irradiância global e a geometria solar (como a mudança do ângulo solar).

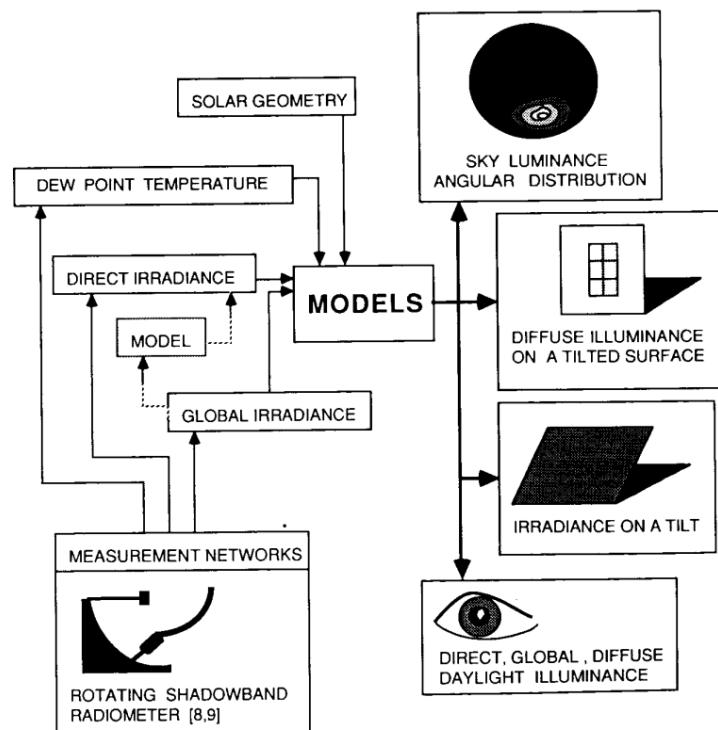


Figura 14 – Relação entre os Inputs e Outputs do Modelo de Perez (PEREZ et al., 1990)

Tabela 2 - RMSE e MBE do Modelo de Perez, Hay e Isotrópico (PEREZ et al., 1990)

Surface Orientation	Mean Global Irradiance (W/m ²)	IRRADIANCE MODEL ERROR (W/m ²)							
		PEREZ 1		ISOTROPIC		HAY		RMSE	MBE
		RMSE	MBE	RMSE	MBE	RMSE	MBE		
90° North	64	11	3	32	18	24	1		
90° East	174	17	-2	43	-7	29	-9		
90° South	230	16	0	38	-17	25	-10		
90° West	173	17	1	43	-6	28	-7		
45° South	396	14	-1	36	-22	22	-11		
Composite error		15	2	39	17	25	9		
ILLUMINANCE MODEL ERROR (100 * LUX)									
Surface Orientation	Mean Global Illumin.	PEREZ 1	PEREZ 2	PEREZ 3	PEREZ 4	ISOTROPIC HAY		RMSE	MBE
		(100*lux)	RMSE	MBE	RMSE	MBE	RMSE		
90° North	71	12	2	12	2	14	6	11	2
90° East	160	17	-1	19	0	19	2	18	-1
90° South	269	16	-1	18	-2	19	-1	19	-2
90° West	173	17	3	20	4	20	6	18	-3
Composite error		16	2	18	2	18	4	17	5
								50	19
								33	7

PEREZ 1: Illuminance coef. PEREZ 2: Dependent Irradiance Coef.
 PEREZ 3: Independent Irradiance Coef.
 ISOTROPIC, HAY, PEREZ 1, 2 and 3: Diffuse Illuminance Input
 PEREZ 4: Operational Model -- Illum. coef. plus diffuse irradiance input --

Ele considera a mesma equação de corrente elétrica que é utilizada no Modelo de Hay e partir dela é possível encontrar o coeficiente de brilho Δ , levando em consideração a massa de ar "m" que é pode ser estimada pelo método desenvolvido por Pickering através da fórmula abaixo.

$$I_0 = G_{sc} \left[1 + 0,33 \cos \left((n-2) \frac{360}{365} \right) \right]$$

$$m = \frac{1}{\cos(Z_s) + 0.50572(96.07995 - Z_s)^{-1.6364}}$$

$$\Delta = \frac{I_{DH} m}{I_0}$$

O índice de claridade é encontrado a partir da equação abaixo, sendo a constante k=1.041:

$$\varepsilon = \frac{\frac{I_{DH} + I_B}{I_{DH}} + k Z_s^3}{1 + k Z_s^3}$$

Após encontrar o índice é necessário verificar a classificação do ambiente de acordo com os limites do coeficiente encontrado, sendo dividido em 8 faixas para ser utilizado nas funções F1 e F2. É importante notar que o limite inferior sempre será 1, já que a irradiação direta na Terra nunca será negativa, entretanto não existe um limite teórico do valor de ε , por isso que qualquer valor superior a 6.2 é considerado como de céu limpo. Entre os dois extremos há outras faixas que significam um meio termo entre nenhuma difusão e muita difusão (e reflexão) dos raios solares através das camadas da atmosfera terrestre.

Tabela 3 - Limites para o coeficiente de clareza (Adaptado, PEREZ et al., 1990)

ϵ Bin	Limite inferior	Limite superior
1 (Nublado)	1.000	1.065
2	1.065	1.230
3	1.230	1.500
4	1.500	1.950
5	1.950	2.800
6	2.800	4.500
7	4.500	6.200
8 (Céu limpo)	6.200	Infinito

A partir do índice acima é necessário encontrar os coeficientes relativos para as 6 subfunções das funções F1 e F2, como demonstrado na tabela abaixo. As funções F1 são relativas ao coeficiente de brilho circunsolar e as funções F2 ao coeficiente de brilho horizontal.

Tabela 4 - Coeficientes de clareza do Modelo de Perez (PEREZ et al., 1990)

ϵBin	F₁₁	F₁₂	F₁₃	F₂₁	F₂₂	F₂₃
IRRADIANCE COEFFICIENTS						
1	-0.008	0.588	-0.062	-0.060	0.072	-0.022
2	0.130	0.683	-0.151	-0.019	0.066	-0.029
3	0.330	0.487	-0.221	0.055	-0.064	-0.026
4	0.568	0.187	-0.295	0.109	-0.152	-0.014
5	0.873	-0.392	-0.362	0.226	-0.462	0.001
6	1.132	-1.237	-0.412	0.288	-0.823	0.056
7	1.060	-1.600	-0.359	0.264	-1.127	0.131
8	0.678	-0.327	-0.250	0.156	-1.377	0.251
ILLUMINANCE COEFFICIENTS						
1	0.011	0.570	-0.081	-0.095	0.158	-0.018
2	0.429	0.363	-0.307	0.050	0.008	-0.065
3	0.809	-0.054	-0.442	0.181	-0.169	-0.092
4	1.014	-0.252	-0.531	0.275	-0.350	-0.096
5	1.282	-0.420	-0.689	0.380	-0.559	-0.114
6	1.426	-0.653	-0.779	0.425	-0.785	-0.097
7	1.485	-1.214	-0.784	0.411	-0.629	-0.082
8	1.170	-0.300	-0.615	0.518	-1.892	-0.055
Circumsolar Brightening Coefficient				$F_1 = F_{11} + F_{12} \cdot \Delta + F_{13} \cdot Z$		
Horizon Brightening Coefficient				$F_2 = F_{21} + F_{22} \cdot \Delta + F_{23} \cdot Z$		

$$F_1(\epsilon) = F_{11}(\epsilon) + F_{12}(\epsilon)\Delta + F_{13}(\epsilon)Z_s$$

$$F_2(\epsilon) = F_{21}(\epsilon) + F_{22}(\epsilon)\Delta + F_{23}(\epsilon)Z_s$$

Depois de calculado para ajustes circunsolares e horizontais, é possível derivar a fórmula final para a irradiância difusa incidente em um coletor de acordo com o Modelo de Perez:

$$I_{DHC} = I_{DH} \left[(1 - F_1) \left(\frac{1 + \cos(\Sigma c)}{2} \right) + F_1 \frac{a}{b} + F_2 \sin(\Sigma s) \right]$$

7.3 Modelo de Liu & Jordan

O modelo de transposição de Liu e Jordan (isotrópico) é uma abordagem matemática utilizada para calcular a radiação solar global em uma superfície inclinada. Este modelo é composto por radiação direta e difusa, bem como radiação refletida do entorno. A componente de radiação direta depende de considerações geométricas, enquanto a radiação difusa depende da distribuição de radiação difusa na atmosfera circundante e do fator de forma da superfície inclinada. A radiação refletida, por outro lado, é uma função das características das superfícies circundantes que são refletores difusos.

O modelo de Liu e Jordan simplifica a distribuição de radiação difusa na atmosfera para ser isotrópica, que é a abordagem mais simples e tem um efeito menor na precisão da previsão da radiação global. Isso ocorre porque a radiação difusa contribui com apenas cerca de 20% da radiação global em céus claros. Ele, porém, é frequentemente usado para determinar a inclinação ótima de um painel solar para maximizar a captação de radiação solar. O modelo leva em consideração a latitude do local, a declinação solar e a hora do dia para calcular a inclinação ótima.

Na formulação matemática é necessário considerar a radiação total em um superfície, sendo $\rho = \text{albedo}$, e os demais componentes equivalentes a radiação diária total, direta, difusa e refletida pelo solo em uma superfície inclinada:

$$\begin{aligned} H_T &= H_{b,T} + H_{d,T} + H_{r,T} \\ H_T &= H_b R_b + H_d \left(\frac{1 + \cos(\beta)}{2} \right) + H \rho \left(\frac{1 - \cos(\beta)}{2} \right) \end{aligned}$$

Esse modelo assume que a intensidade da radiação difusa é uniforme no topo da órbita acima a atmosférica, assumindo ser isotrópica, consequentemente. R_b é a fração da radiação direta diária em um plano inclinado relativo a um plano horizontal. No hemisfério norte é dada pela equação:

$$R_b = \frac{\cos(\varphi - \beta) \cos(\delta) \sin(\omega'_s) + \omega'_s \sin(\varphi - \beta) \sin(\delta)}{\cos(\varphi) \cos(\delta) \sin(\omega'_s) + \omega'_s \sin(\varphi) \sin(\delta)}$$

Dentre os termos acima φ , δ e ω'_s representam respectivamente a latitude, a declinação e o ângulo solar do pôr do sol para uma superfície horizontal, sendo esse último de acordo com a fórmula (em radianos):

$$\omega_s = \cos^{-1}(-\tan(\varphi)\tan(\delta))$$

Para uma superfície inclinada:

$$\omega'_s = \min(\cos^{-1}(-\tan(\varphi)\tan(\delta)), \cos^{-1}(-\tan(\varphi - \beta)\tan(\delta)))$$

Considerando as equações acima, o índice de claridade K_T é dado pela divisão entre a radiação global horizontal e a radiação extraterrestre em uma superfície horizontal:

$$K_T = \frac{H}{H_0}$$

A radiação extraterrestre em uma superfície horizontal pode ser calculada como, considerando j_d como o dia no calendário Juliano:

$$H_0 = \frac{24}{\pi} H_{sc} \left[1 + 0.033 \cos\left(\frac{2\pi j_d}{365}\right) \cos(\varphi) \cos(\delta) \sin(\omega_s) + \omega_s \sin(\varphi) \sin(\delta) \right]$$

Fora da atmosfera praticamente não há radiação difusa ou albedo, então H_0 é considerando composto apenas por radiação direta. Para superfícies inclinadas somente H_{T0} é necessário e de acordo com a relação de H_{bT} , H_{T0} pode ser calculada.

$$H_{bT} = H_b R_b$$

$$H_{T0} = H_0 R_b$$

Seria interessante abordá-lo e realizar uma aplicação do código proposto junto aos demais modelos, entretanto ele não está disponível nos softwares PVsyst e Helioscope, somente os modelos de transposição de Perez e Hay-Davies.

8. Softwares de simulação

Os softwares de simulação PVsyst, SAM e HelioScope são amplamente utilizados na indústria solar para análise e dimensionamento de sistemas fotovoltaicos. Cada um desses softwares possui características e funcionalidades distintas que os tornam úteis em diferentes aplicações, sendo o PVsyst o mais utilizado dentre os 3 softwares e com mais recursos para as simulações. O HelioScope pode ser considerado o mais user-friendly, com forte apelo de desenvolvimento de projetos não só pelo ponto de vista físico, mas também do ponto de vista financeiro, considerando o sistema de energia solar como um investimento e que possui um retorno considerando a sua geração de energia.

8.1 PVsyst

O PVsyst é um software de simulação de sistemas fotovoltaicos amplamente reconhecido pela sua precisão e recursos avançados. Ele permite modelar e simular o desempenho de sistemas fotovoltaicos em diferentes cenários, levando em consideração fatores como a radiação solar, a geometria dos módulos, o sombreamento e as perdas do sistema. O PVsyst é amplamente utilizado para análise de viabilidade de projetos solares, otimização de sistemas e estimativa de produção de energia.

O projeto necessita de uma localização geográfica e alguma base de dados meteorológica em sua concepção. A seguir, define-se variáveis básicas do sistema como a orientação dos módulos fotovoltaicos, a potência desejada ou área e o tipo de módulos e inversores. O PVsyst vai propor uma configuração básica e definir valores padrão razoáveis para todos os parâmetros necessários para os cálculos, definindo uma primeira variante. Em seguida, é possível definir perturbações como sombras distantes, sombras próximas, parâmetros de perdas específicos, avaliação econômica dentre outras.

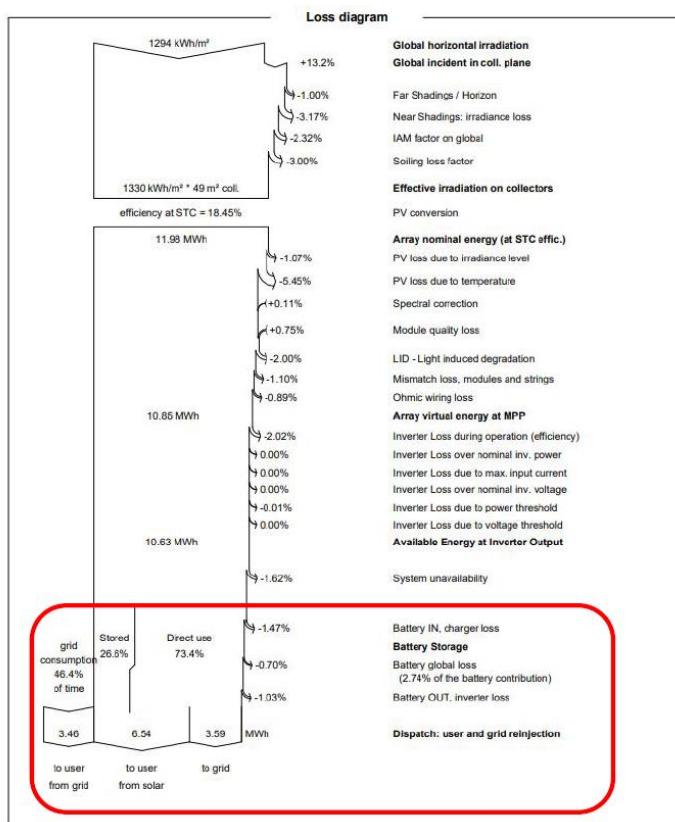


Figura 15 - Diagrama de perdas do PVsyst (PVsyst7, s.d.)

8.2 SAM

Apesar de não ser utilizado para a simulação porque foram encontrados valores incoerentes do plane of array para o local selecionado, O SAM (System Advisor Model), desenvolvido pela National Renewable Energy Laboratory (NREL), é uma ferramenta poderosa para avaliação econômica e análise técnica de projetos solares. De acordo com a própria descrição no General Description mais atualizado (2017), ele é modelo de software técnico-econômico com o intuito de facilitar a tomada de decisões para pessoas inseridas na indústria de energia renovável. O software abrange uma ampla gama de tecnologias, incluindo sistemas fotovoltaicos, sistemas de concentração solar, sistemas de energia eólica e sistemas de armazenamento de energia. O SAM permite a análise de viabilidade financeira, otimização de projetos e avaliação de desempenho energético.

Para o projeto é necessário escolher um modelo de performance e um modelo financeiro para o projeto, além de definir valores de input como o local do projeto, tipo de equipamento utilizado, custo de instalação e de operação do sistema e incentivos financeiros. Após definir as variáveis de entrada é necessário executar as simulações e examinar os resultados.



Figura 16 - Interface do SAM (SAM General Description, 2017)

Ele foi desenvolvido em uma parceria do NREL com o Sandia National Laboratories em 2005 para uso interno do Departamento de Energia dos Estados Unidos (DOE), até sua primeira versão pública em Agosto de 2007. Desde então, o NREL continua fornecendo o desenvolvimento e distribuição gratuita do software.

8.3 HelioScope

O HelioScope, diferente dos outros 2 anteriores, é um software baseado em nuvem para o design e análise de sistemas solares. Ele oferece uma interface intuitiva e recursos avançados para dimensionamento, modelagem de sombreamento e estimativa de produção de energia. O HelioScope permite a criação de projetos solares precisos, levando em consideração fatores como a topografia do terreno, a inclinação dos módulos, o sombreamento e outras variáveis relevantes.

O software é projetado para simular o output de um arranjo fotovoltaico a partir da premissas tecnológicas, de arquitetura e do meio ambiente. A simulação é alcançada através de um modelo de sistema que simula todos os componentes elétricos do arranjo individualmente, permitindo que aconteça uma interação entre eles de forma realista, permitindo o modelo computar perdas e dinâmicas da performance a partir de certos princípios básicos, ao contrário de modelos que usam um único modelo de performance para assumir o output escalando a partir da área disponível (no caso, não é referente ao PVsyst e nem ao SAM). O HelioScope também adiciona outros fatores típicos de perda como perdas em linha (transmissão), mismatch (o descompasso existente entre módulos conectados em série ou paralelo em virtude de fatores extrínsecos ou intrínsecos ao sistema fotovoltaico (SAKÔ et al., 2019)) e sombreamento por exemplo.

Tabela 5 - Perdas típicas no HelioScope (HelioScope 101, 2023)

⚡ Annual Production			
	Description	Output	% Delta
Irradiance (kWh/m ²)	Annual Global Horizontal Irradiance	2,089.6	
	POA Irradiance	2,239.5	7.2%
	Shaded Irradiance	2,217.5	-1.0%
	Irradiance after Reflection	2,150.2	-3.0%
	Irradiance after Soiling	2,107.2	-2.0%
	Total Collector Irradiance	2,107.2	-0.0%
Energy (kWh)	Nameplate	2,523,218.5	
	Output at Irradiance Levels	2,510,902.9	-0.5%
	Output at Cell Temperature Derate	2,311,664.2	-7.9%
	Output After Mismatch	2,217,896.5	-4.1%
	Optimizer Output	0.0	0.0%
	System DC Output	2,198,218.5	-0.9%
System AC Output		2,133,524.3	-2.9%

9. Incerteza associada a meteorologia

A meteorologia é uma ciência que lida com a previsão e o estudo do tempo atmosférico e desempenha um papel crucial na avaliação e estimativa da radiação solar em projetos de energia solar. No entanto, é importante reconhecer que existem incertezas associadas às previsões meteorológicas e aos dados meteorológicos utilizados para estimar a radiação solar. Essas incertezas podem impactar a precisão das análises e projeções realizadas.

Uma das principais fontes de incerteza na meteorologia é a própria natureza complexa e dinâmica da atmosfera. A atmosfera é afetada por uma variedade de fatores, como mudanças climáticas, padrões de circulação atmosférica e fenômenos meteorológicos de curto prazo, como nuvens, chuvas e nevoeiros. A capacidade de prever com precisão esses fenômenos e sua influência na radiação solar podem ser desafiadores. Além disso, a disponibilidade e a qualidade dos dados meteorológicos podem ser uma fonte de incerteza. Os dados coletados por estações meteorológicas e satélites podem conter erros de medição, lacunas temporais ou espaciais e variações na calibração dos instrumentos. A densidade da rede de estações meteorológicas pode variar em diferentes regiões, o que por vezes afeta a precisão dos dados utilizados para estimar a radiação solar.

As incertezas na meteorologia também estão relacionadas às próprias limitações dos modelos de previsão meteorológica. Esses modelos são baseados em equações matemáticas que descrevem o comportamento da atmosfera, mas eles incorporam simplificações e suposições que podem introduzir incertezas nas previsões. Além disso, a precisão das previsões de longo prazo tende a diminuir à medida que o horizonte temporal aumenta, por isso que são utilizados dados climáticos de anos anteriores para estimar os dados futuros, considerando uma certa estabilidade e previsibilidade em determinados parâmetros.

O NSRDB SUNY Semi-Empirical Model do NREL usa o modelo semiempírico de dados de satélite (PEREZ et al., 2002, 2004, 2013) desenvolvido como parte de um diálogo (acordo ou tratado) de energia entre os Estados Unidos e a Índia, em parceria com a State University of New York at Albany. Esse modelo converte um “índice de nuvem” dos satélites para um índice de claridade, que é usado em seguida para output do modelo de céu limpo SOLIS para o GHI. Para o cálculo do DNI e DHI o GHI é utilizado no modelo de DIRINT. Entretanto o SUNY só está disponível na região sul da Ásia, por isso o modelo utilizado de dados é o PSM V3 (Physical Solar Model).

A NREL possui tanto estações meteorológicas em solo como as bases meteorológicas em satélite, sendo que a partir das imagens em alta resolução dos satélites e a análise dos pixels, aplicando coeficientes ou índices de nebulosidade (cloud index) é possível determinar parâmetros como a irradiação global horizontal dentre outros. Muitas vezes os valores fornecidos pelos satélites são mais fiéis do que os registrados em solo, especialmente em regiões muito esparsas onde há poucas estações meteorológicas disponíveis, especialmente em áreas inhóspitas ao ser humano como desertos, florestas ou nas regiões polares do planeta.

Para a validação do código a estação escolhida foi a que é utilizada normalmente e está entre as disponíveis na biblioteca pvlib, sendo classificada como classe I pela NREL, sendo as de classe III com baixa precisão e muitos dados faltantes, a II uma categoria intermediária e as de classe I as melhores para realizar simulações matemáticas. Apesar de não necessariamente possuírem todos os valores acurados ao longo do ano, ao menos a NREL certifica que não possuem dados omissos, por isso no código no final do trabalho não foi analisado esse item.

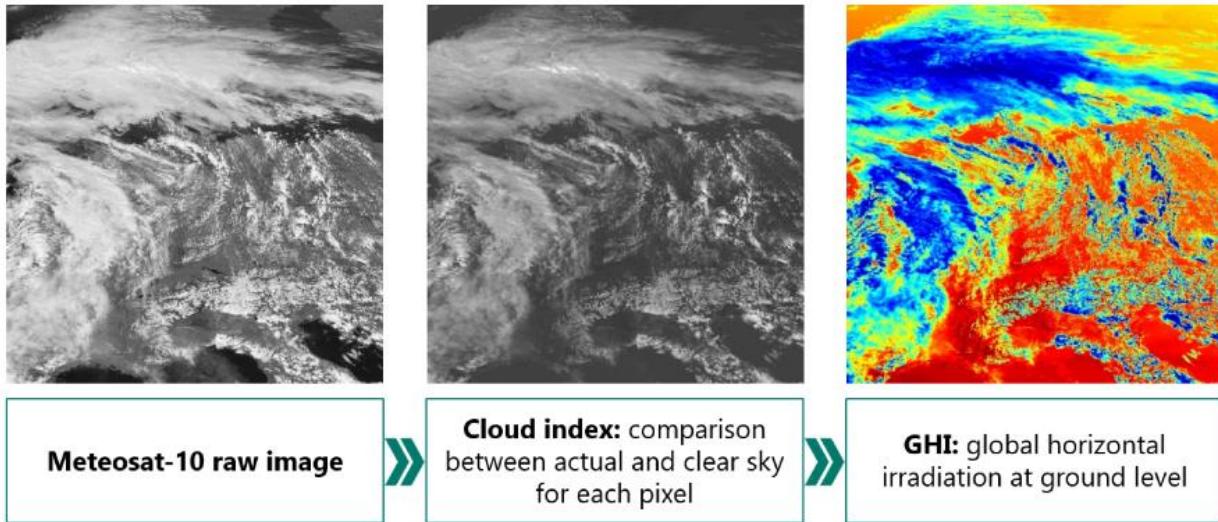


Figura 17 - Arquivo meteorológico de satélite (HELIOSCOPE, 2023)

9.1 Fenômenos cíclicos

A meteorologia estuda os fenômenos atmosféricos e climáticos que ocorrem na Terra, e muitos desses fenômenos exibem padrões cíclicos. Esses padrões são influenciados por uma série de fatores, como a inclinação axial da Terra, as correntes oceânicas e os ciclos solares. Esses fenômenos cíclicos são de grande importância para entender e prever as condições climáticas e a capacidade de identificar e monitorar esses padrões cíclicos resulta em previsões mais precisas. Além disso, o seu estudo ajuda a compreender melhor as tendências climáticas de longo prazo.

Dois exemplos de fenômenos cíclicos que são acoplados (atmosférico-oceânico) na meteorologia ocorrem no Oceano Pacífico equatorial, sendo caracterizados como ENOS ou El Niño Oscilação Sul (El Niño) e na sua fase oposta, La Niña. Esses eventos são caracterizados por anomalias na temperatura da superfície do mar e têm um impacto significativo no clima global, afetando o transporte de umidade, a precipitação, os padrões de vento e as temperaturas em diferentes regiões do mundo.

Enquanto o primeiro é caracterizado pelo aumento da temperatura média normal histórica, no La Niña o efeito é o oposto, com o resfriamento em relação às condições normais. Dependendo da época do ano que ela ocorre, assim como no El Niño, as mudanças ocorrem em diferentes áreas do globo, especialmente a porção pacífica da América do Sul, da Oceania e da Ásia, região atlântica da América do Sul, América Central e às vezes parte da África, América do Norte e áreas banhadas pelo Oceano Índico.

Anomalia de Temperatura da Superfície do Mar 22/06/2023 a 29/06/2023

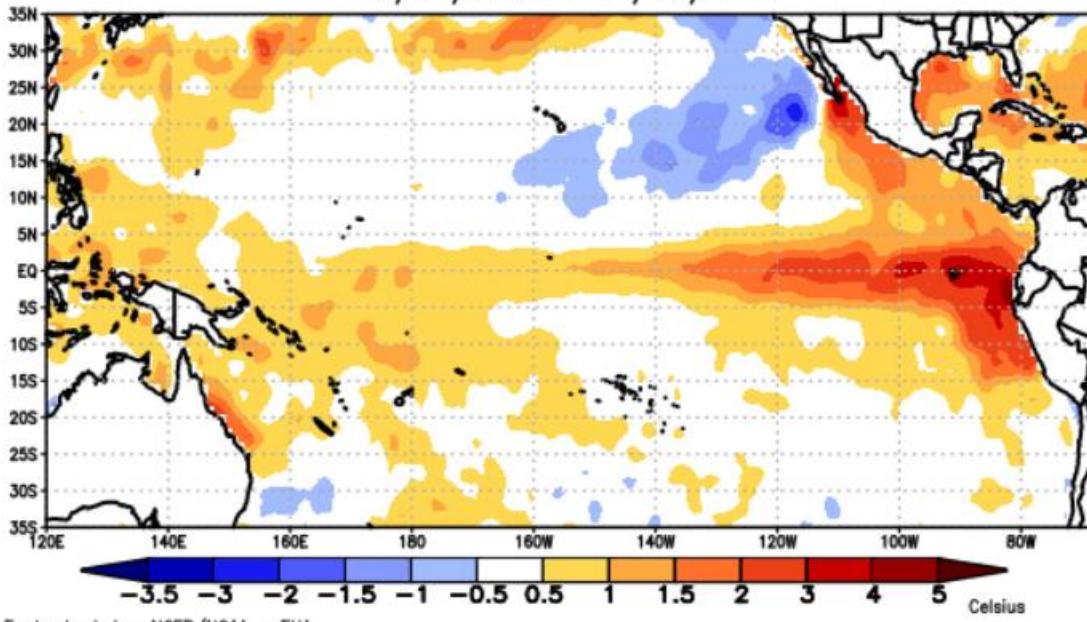


Figura 18 - Anomalia de temperatura devido ao El Niño em 2023 (INPE, 2023)

Além disso, o ciclo solar de aproximadamente 11 anos é um fenômeno cíclico que afeta a atividade solar e, consequentemente, a radiação solar recebida pela Terra. Durante o ciclo solar, a quantidade de manchas solares – regiões do Sol com menor temperatura e que são menos brilhantes – e a atividade magnética do Sol variam com um certo padrão cíclico que são identificados por diferentes metodologias como o estudo de isótopos radioativos na Terra, que são afetados pela atividade solar.

Durante o máximo solar há um aumento no número de manchas solares, entretanto a atividade solar é mais alta no período, enquanto no mínimo solar a quantidade de manchas solares é menor e sua atividade é alta, com cada ciclo solar com duração aproximada de 11 anos, seja o ciclo de máximo ou ciclo de mínimo. Não é somente o clima que é afetado por esses ciclos, é possível observar interrupções em comunicações via satélite em períodos de máximo e inclusive aumento na radiação que os astronautas, aeronaves e equipamentos recebem.

Um estudo publicado na revista "Solar Physics" (BREHM et al., 2021) usou radionuclídeos cosmogênicos para investigar a existência e persistência de ciclos solares de 11 anos ao longo do último milênio. Os autores descobriram que os ciclos solares de 11 anos têm sido uma característica persistente da atividade solar ao menos nos últimos 1000 anos sugerindo que, ao menos, em um período de 1 milênio esse comportamento é persistente.

9.2 Eventos desconhecidos

A meteorologia é uma ciência complexa que busca entender e prever os fenômenos atmosféricos, mas ainda existem muitos aspectos desconhecidos e fenômenos que desafiam nossa compreensão. Por mais que em alguns deles algumas hipóteses são promissoras para sua explicação, ainda não são totalmente validados no meio acadêmico. A natureza complexa e interconectada do clima e da atmosfera da Terra pode levar a fenômenos que desafiam nossa compreensão atual.

Meteotsunamis são tsunamis que são desencadeados por eventos meteorológicos, ao invés de atividade sísmica. Eles são geralmente causados por sistemas de tempestade de movimento rápido que criam ondas de pressão atmosférica. Um estudo de (OKAL, 2020) explorou a possibilidade de registrar meteotsunamis usando componentes horizontais de sismômetros baseados em terra perto de costas. O estudo sugere que os registros sísmicos poderiam desempenhar um papel importante na compreensão da estrutura dos meteotsunamis.

9.3 Equipamentos para medição

A medição precisa das variáveis meteorológicas desempenha um papel fundamental na meteorologia e na avaliação de recursos solares. Para isso, uma variedade de equipamentos é utilizada para coletar dados meteorológicos e solares.



Figura 19 – Foto de uma das estações de coleta de dados de radiação (Atlas brasileiro de energia solar 2º Ed, 2017)



Figura 20 – Estações meteorológicas automáticas (EMA's) (Atlas brasileiro de energia solar 2º Ed, 2017)

O piranômetro é um tipo de radiômetro solar que é usado – em sua concepção original – para medir a irradiação solar global, que é a soma da irradiação direta, difusa e refletida pelas superfícies ao redor do sensor. O piranômetro é projetado para ter uma resposta espectral que é próxima à do olho humano, o que significa que ele é mais sensível à luz visível e menos sensível à luz ultravioleta e infravermelha. Os

piranômetros são usados em uma variedade de aplicações, incluindo a medição da radiação solar para a geração de energia solar, a pesquisa climática e a previsão do tempo.

O piranômetro pode ser composto por um sensor termopilha que absorve a radiação solar e a converte em calor. Esse calor é então convertido em um sinal elétrico que pode ser medido. O sensor termopilha é coberto por uma cúpula de vidro que permite a passagem da radiação solar. A cúpula também ajuda a minimizar os efeitos do vento e da chuva no sensor. Outro modelo de piranômetro é o baseado em fotodiodo, que não necessita da etapa de conversão do calor para sinal elétrico, já sendo intrínseco a sua resposta aos fótons do Sol. O fotodetector é capaz de converter a luz em corrente ou tensão. Quando a luz solar atinge o fotodiodo, ela excita os elétrons e cria pares de elétrons-buracos, que geram uma corrente elétrica proporcional à intensidade da luz, usando a mesma concepção das células solares utilizadas em módulos fotovoltaicos.



Figura 22. Piranômetro de fotodiodo de silício.
Fonte: Kipp&Zonen (2016).

Figura 21 – Piranômetro (Atlas brasileiro de energia solar 2º Ed, 2017)



Figura 22 - Piranômetro de fotodiodo e seu leitor (Autoria própria, 2023)

O pireliômetro é um instrumento que é usado para medir a irradiação solar direta. Difere do piranômetro na medida em que tem um campo de visão muito estreito e deve ser apontado diretamente para o sol para obter uma leitura precisa. Ele é composto por um tubo longo, no final do qual está um sensor termopilha. O tubo ajuda a bloquear a luz difusa e refletida, permitindo que apenas a luz direta do sol atinja o sensor. Como o piranômetro de termopilha, o pireliômetro converte a radiação solar em calor, que é então convertido em um sinal elétrico que pode ser medido após uma conversão determinada pelo seu projeto. Os pireliômetros baseados em fotodiodos são geralmente menores, mais leves e mais baratos do que os pirheliômetros baseados em termopilhas e seu funcionamento é semelhante ao piranômetro descrito acima, através da excitação dos elétrons e a criação de pares elétrons-buracos. No

entanto, eles também são mais sensíveis às variações de temperatura e podem ter uma resposta espectral menos precisa.

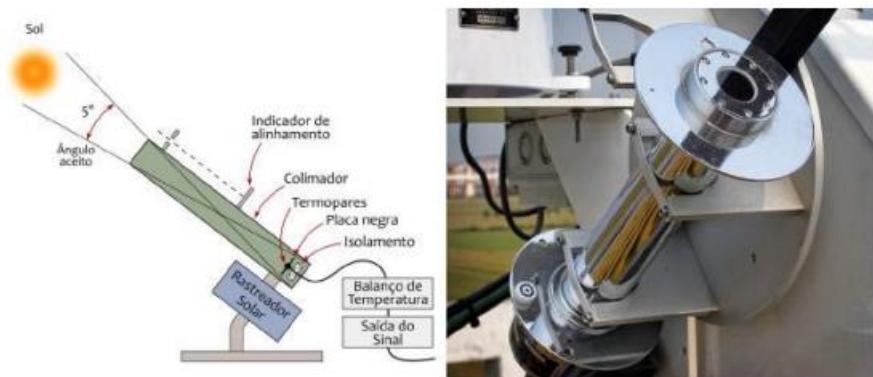


Figura 23 – Pireliômetro (Atlas brasileiro de energia solar 2º Ed, 2017)

Os pirômetros são instrumentos especializados usados para medir a radiação de onda longa ou radiação infravermelha que é emitida pela Terra e pela atmosfera. Essa medição é fundamental para uma variedade de aplicações, incluindo estudos climáticos, meteorologia e modelagem de sistemas de energia solar. Eles são projetados para ter uma resposta espectral que é sensível à radiação infravermelha na faixa de 4 a 50 micrômetros. São usados em uma variedade de aplicações, incluindo a medição da radiação de onda longa para estudos climáticos, a previsão do tempo e a modelagem da eficiência dos sistemas de energia solar.

Os pirômetros baseados em termopilhas são a forma mais comum de pirômetro. Eles são compostos por um sensor termopilha que absorve a radiação infravermelha e a converte em calor que é então convertido em um sinal elétrico que pode ser medido. Os pirômetros baseados em termopilhas são geralmente robustos, precisos e capazes de operar em uma ampla gama de condições ambientais; no entanto, eles também podem ser relativamente caros e podem requerer calibração regular para garantir a precisão. Os pirômetros baseados em fotodiodes são uma alternativa mais recente aos pirômetros baseados em termopilhas, são compostos por um fotodiode que é sensível à radiação infravermelha. Quando a radiação infravermelha atinge o fotodiode, ela excita os elétrons e cria pares de elétrons-buracos, que geram uma corrente elétrica ou tensão proporcional à intensidade da radiação, da mesma forma que os demais sensores citados anteriormente. Os pirômetros baseados em fotodiodes são geralmente menores, mais leves e mais baratos, porém eles também são mais sensíveis às variações de temperatura e podem ter uma resposta espectral menos precisa.

Além das estações meteorológicas terrestres, os satélites meteorológicos desempenham um papel crucial na coleta de dados atmosféricos em escala global. Os satélites meteorológicos, como os da série GOES (Geostationary Operational Environmental Satellites) operados pela NOAA (National Oceanic and Atmospheric Administration) dos EUA, e os satélites Meteosat operados pela EUMETSAT (European Organisation for the Exploitation of Meteorological Satellites), fornecem dados contínuos sobre a temperatura da superfície da Terra, a cobertura de nuvens, a irradiância solar que incide sobre a Terra, além da irradiância refletida pelas nuvens e superfície, assim como outros parâmetros atmosféricos através de análise de suas imagens e de análise de ondas eletromagnéticas, principalmente.

10. Aprofundamento na regressão não-linear

A regressão, em suma, é dividida em um primeiro momento para funções lineares e funções não-lineares, sejam criadas para uma análise com equações bem definidas ou para, primordialmente, lidar com o tratamento de dados, não só da natureza e sim, atualmente, aplicações de big data, machine learning e deep learning. Caso a aplicação deste estudo fosse para a irradiação extraterrestre do sol a regressão linear poderia ser interessante, visto que a variação da irradiância ao longo de um dia é mínima para um ponto no espaço fixo e a distância da Terra não varia em escalas astronômicas, mesmo considerando sua trajetória elíptica e a diferença em "ua" (unidade astronômica) entre o periélio e afélio da órbita terrestre. Ao lidar com a irradiância em um plano na terra, seja a nível do mar ou em montanhas de milhares de metros de extensão, o comportamento já é totalmente diferente levando em conta apenas a translação e rotação do planeta, sem contar os fenômenos climáticos que são de conhecimento da ciência e os que ainda não há definições claras sobre o seu comportamento. Por conta da premissa anterior torna-se importante utilizar a regressão não-linear, entretanto os dois métodos serão expostos abaixo.

$$y = \beta_0 + \beta_1 x + \epsilon$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_n x_n + \epsilon$$

$$y = A \sin(B(x - C)) + D + \epsilon$$

A regressão pode ser modelada a partir de diferentes tipos de funções, como as lineares de 1 ou n variáveis acima ou como a senoidal por exemplo (que foi escolhida para este trabalho), sendo o ϵ o relativo ao erro aleatório.

10.1 Regressão linear

É uma técnica estatística amplamente utilizada que busca entender a relação entre duas ou mais variáveis. A ideia básica é determinar a melhor linha de ajuste (ou plano em dimensões acima de duas) que descreve a relação entre as variáveis. A regressão linear pode ser simples, com uma variável independente, ou múltipla, com várias variáveis independentes. Ela é frequentemente usada em aprendizado de máquina e modelagem estatística para prever resultados com base em variáveis independentes, a partir do ajuste de uma equação linear em relação aos dados observados. Para realizar uma regressão linear é necessário estimar os coeficientes da equação linear, dependendo da quantidade de variáveis escolhidas.

Para funções lineares no tempo é possível utilizar a regressão linear para o seu tratamento, não sendo muito utilizada para aplicações de fenômenos da natureza pelo – na maioria dos casos – comportamento não-linear. Apesar desta restrição inicial, ainda assim existe uma forma de utilizá-la: caso exista uma linearização dos dados a regressão linear pode ser utilizada, o que pode ser interessante para determinadas situações em que a regressão não-linear possua um custo computacional excessivo para lidar com os dados reais, por mais que eles não sejam lineares.

A regressão linear é uma técnica poderosa, mas tem suas limitações. Como ela assume uma relação linear entre as variáveis, o que nem sempre é o caso, pode ter como resultado uma função que não fará sentido na prática. Além disso, pode ser afetada por outliers e pode sofrer de multicolinearidade (quando as variáveis independentes estão altamente correlacionadas entre si), portanto não foi escolhida para o código de correção das curvas de irradiância em um plano.

10.2 Regressão não-linear

A regressão não linear é um método de modelagem que permite a relação entre variáveis independentes e uma variável dependente ser modelada por uma função não linear. Este método é amplamente utilizado em muitos campos devido à sua flexibilidade para descrever principalmente complexas relações não lineares que são muito complicadas para identificar com outros métodos matemáticos.

Um exemplo de regressão não linear é a regressão de mínimos quadrados não lineares, que será utilizada no código para o tratamento de outliers de irradiação solar para um plano inclinado sobre a superfície terrestre. Este método é uma extensão do método de mínimos quadrados lineares e é usado para modelar relações não lineares entre variáveis. A ideia básica por trás da regressão de mínimos quadrados não lineares é minimizar a soma dos quadrados das diferenças entre os valores observados e os valores previstos pela função não linear. Este método é frequentemente usado em problemas de otimização de uma forma geral e estimação de parâmetros desconhecidos, como os parâmetros de uma função senoide que melhor descreve a tendência de irradiação diária.

O algoritmo Levenberg-Marquardt é um algoritmo amplamente utilizado para a solução de problemas de mínimos quadrados não lineares, sendo que ele combina os métodos de Gauss-Newton e o gradiente descendente a fim de fornecer uma solução robusta e eficiente para problemas de mínimos quadrados não lineares. O algoritmo é particularmente útil quando a função objetivo tem muitos mínimos locais, pois é capaz de escapar de mínimos locais e encontrar o mínimo global por conta do algoritmo de gradiente descendente, muito utilizado em aplicações de machine learning, por conta de muitas vezes possuírem muitos mínimos locais. Apesar desta vantagem, na seção seguinte será demonstrado que - para evitar um tempo computacional muito elevado – todos os valores de irradiação que não possuem nenhum valor de potência não entram na regressão não-linear, já que cerca de metade dos valores ao longo de um dia em latitudes não muito extremas são iguais a 0 W/m², tornando-os mínimos globais e por vezes mínimos locais (já que na maioria dos dias, considerando valores horários de irradiação, as funções são convexas). Entretanto, caso essa restrição não fosse aplicada o algoritmo de Levenberg-Marquardt seria adequado também, considerando esta vantagem que ele possui em relação aos demais.

Não obstante, a regressão não linear também pode ser combinada com algoritmos denominados meta-heurísticos para melhorar ainda mais a precisão da modelagem. Por exemplo, o algoritmo de otimização de enxame de partículas (PSO) e o algoritmo genético (GA) por vezes são utilizados em combinação com a regressão não linear para resolver problemas complexos de otimização e modelagem, porém não haverá tal combinação no código por tratar-se de um código piloto, com foco na aplicação pura do método de regressão, a não ser quando for necessário tratar valores incoerentes de irradiação que são muito maiores que valores típicos de irradiação na Terra.

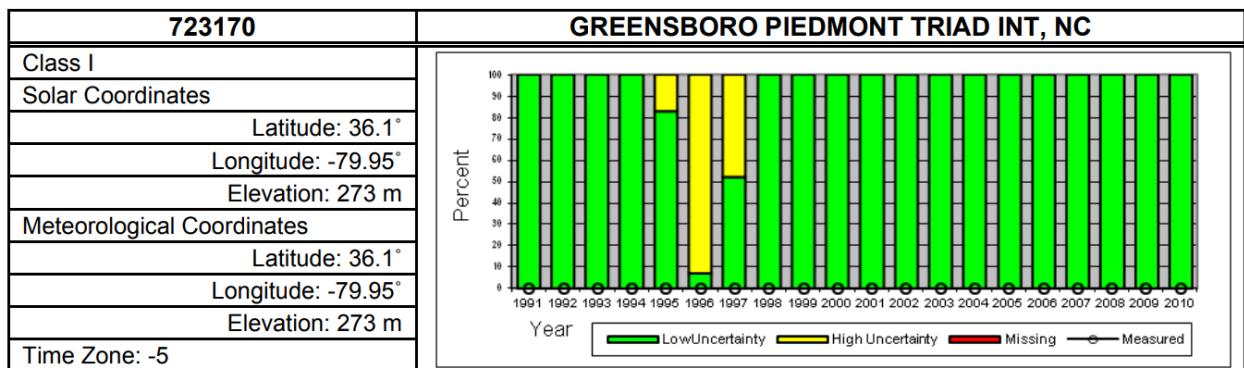
11. Algoritmo para modelagem de outliers de irradiância solar

A principal fonte meteorológica utilizada foi da estação meteorológica do aeroporto Piedmont Triad International Airport, em Guilford County, próximo a Greensboro, Carolina do Norte. O arquivo TYM3 é de uma estação "classe I" de acordo com a NREL, com menor nível de incerteza. Além disso, foram comparados 24 anos para escolher o ano com valores mais típicos da região, conforme a tabela abaixo retirada de um documento da NREL. Para comparar os valores dos códigos em Python uma simulação no HelioScope foi realizada, utilizando o mesmo conjunto meteorológico usualmente recomendado pelo pvlib e outra fonte, de satélite, da região.

Tabela 6 - Classificação da estação de Piedmont Triad (NREL, 2008)

USAF	Station Name	State	Latitude	Longitude	Time Zone	Elevation	NSRDB Class	Pool Years
723030	FAYETTEVILLE POPE AFB	NC	35.167	-79.017	-5	66	II	12
723035	FAYETTEVILLE RGNL G	NC	34.983	-78.883	-5	59	II	12
746930	FORT BRAGG SIMMONS AAF	NC	35.133	-78.933	-5	93	II	12
723066	GOLDSBORO SEYMOUR JOHNSON AFB	NC	35.350	-77.967	-5	33	II	12
723170	GREENSBORO PIEDMONT TRIAD INT	NC	36.100	-79.950	-5	273	I	24

Tabela 7 - Incerteza da estação de Piedmont Triad ao longo dos anos (NREL, 2012)



11.1 Regressão não-linear com o Algoritmo Levenberg–Marquardt

A biblioteca scipy possui, no pacote A função curve_fit do scipy.optimize realiza uma técnica chamada regressão não linear de mínimos quadrados. Matematicamente, a ideia é minimizar a soma dos quadrados dos resíduos, onde um resíduo é a diferença entre um valor observado e um valor previsto pela função que está sendo ajustada.

A regressão não linear de mínimos quadrados busca encontrar os parâmetros p que minimizam S. Isso é feito através de um processo iterativo que começa com uma estimativa inicial para p e gradualmente refina essa estimativa até que S não possa ser reduzido significativamente. Ela é uma técnica poderosa que pode ajustar uma ampla variedade de funções aos dados, mas também pode ser sensível à estimativa inicial de p. Se a estimativa inicial estiver muito longe do verdadeiro valor de p, o processo iterativo pode não convergir para a solução correta.

Para minimizar a função objeto S , da soma dos quadrados dos resíduos, é necessário calcular o vetor dos resíduos r_i , que é formado pela diferença entre o i -ésimo valor observado y_i e o i -ésimo valor previsto da função f com a variável independente x_i e os parâmetros p .

$$r_i = y_i - f(x_i, p)$$

Após os cálculos dos vetores é realizada a soma dos quadrados dos resíduos para criar a função objetivo S .

$$S = \sum r_i^2$$

O Algoritmo de Levenberg–Marquardt em si busca encontrar os parâmetros p que minimizam a função objetivo S através de um processo iterativo, começando com uma estimativa inicial para os parâmetros e gradualmente ajustando até que S possa ser reduzido significativamente. A cada iteração a atualização dos parâmetros é dada pela solução do sistema linear composto pelas matrizes jacobianas e jacobianas transpostas J de f com respeito a p e calculada em x_i e p , o parâmetro de amortecimento λ , a matriz identidade I , o delta dos parâmetros p Δp e o vetor de resíduos r . λ é ajustado a cada iteração, sendo que se a atualização diminui a função objetivo S o valor de λ também é diminuído, aproximando mais ao método de Gauss–Newton, caso o contrário o valor de λ é aumentado, com o algoritmo se comportando mais como o método de gradiente descendente.

$$(J^T J + \lambda I) \Delta p = J^T r$$

11.2 Medidas estatísticas e de análise de sinal

11.2.1 Assimetria

A assimetria é uma medida estatística que descreve a distribuição dos dados em relação à sua média. Em termos simples, a assimetria indica o grau de desvio da simetria em uma distribuição de probabilidade. Se a distribuição é simétrica, a assimetria é zero. Uma distribuição com uma cauda longa à direita (valores maiores) tem assimetria positiva, enquanto uma distribuição com uma cauda longa à esquerda (valores menores) tem assimetria negativa. A assimetria é calculada como o terceiro momento padronizado dos dados, ou seja, a média dos cubos dos desvios padrão, fornecendo informações valiosas sobre onde a maioria dos valores está concentrada em uma distribuição.

Em aplicações práticas, a assimetria pode ser usada para identificar outliers ou melhorar modelos de previsão, nesse trabalho a intenção é realizar uma comparação antes e após o tratamento de outliers nas curvas de irradiância diárias. Na fórmula abaixo x_i são os dados, \bar{x} é a média, n é a quantidade e s o desvio padrão deles.

$$\text{Assimetria} = \frac{\frac{\sum_{i=1}^n (x_i - \bar{x})^3}{n}}{s^3}$$

11.2.2 Envelope

A Transformada de Hilbert é uma ferramenta matemática usada para manipular funções de sinal e é frequentemente usada em processamento de sinal e análise de dados. O envelope de um sinal, sua amplitude (magnitude) instantânea, pode ser obtido usando a Transformada de Hilbert, que gera uma

curva suave que envolve o sinal original e captura suas variações de amplitude. A partir de um sinal $x(t)$, sua Transformada de Hilbert ($\bar{x}(t)$) é dada pela expressão abaixo, onde o segundo termo é a componente imaginária multiplicada pela Transformada de Hilbert.

$$\bar{x}(t) = x(t) + jH[x(t)]$$

O envelope do sinal $x(t)$ é o módulo do sinal resultante da transformada, o sinal analítico $\bar{x}(t)$, com a expressão fornecida abaixo:

$$E(t) = |x(t) + jH[x(t)]|$$

A média dos valores do envelope da Transformada de Hilbert pode ser interpretada como uma medida da amplitude média do sinal original ao longo do tempo, pode ser vista como uma medida do "nível" médio do sinal. Se a média do envelope é alta, isso indica que o sinal tem uma amplitude média alta. Se a média do envelope é baixa, isso indica que o sinal tem uma amplitude média baixa. O envelope pode possuir sua assimetria, curtose e coeficiente de variação avaliados também, entretanto neste estudo somente sua média foi considerada, por não se aprofundar tanto na análises de sinal, porque o seu intuito não é esse.

11.2.3 Curtose

A curtose é uma medida estatística que descreve o "achatamento" ou "pico" de uma distribuição de probabilidade. É possível dizer que a curtose mede a quantidade de dados próximos à média em comparação com os dados distantes da média. Uma distribuição com curtose alta tem um pico mais agudo e caudas mais pesadas, enquanto uma distribuição com curtose baixa é mais achatada com caudas mais leves. A curtose é calculada como o quarto momento padronizado dos dados. Assim como a assimetria, a curtose é uma ferramenta útil para identificar outliers e entender a estrutura dos dados. Em aplicações financeiras, por exemplo, a curtose é frequentemente usada para medir o risco de eventos extremos, que possuam outliers com valores muito acima ou abaixo do esperado, portanto é interessante utilizá-la também para comparações.

$$Curtose = \frac{\sum_{i=1}^n (x_i - \bar{x})^4}{\frac{n}{s^4}} - 3$$

11.2.4 Coeficiente de variação

O coeficiente de variação (CV) é uma medida estatística que descreve a variabilidade dos dados em relação à média. É expresso como uma porcentagem e é calculado como o desvio padrão dividido pela média, multiplicado por 100 (para possuir um valor percentual). O CV é uma medida de dispersão relativa, o que significa que ele permite comparar a variabilidade entre conjuntos de dados que têm unidades de medida diferentes ou médias muito diferentes. Em muitas disciplinas, incluindo finanças, ecologia e engenharia, o CV é uma ferramenta valiosa para comparar a variabilidade de diferentes conjuntos de dados em um terreno comum, portanto para o caso da irradiância solar ao longo do ano, que pode ter o valor de irradiância categorizado por diferentes conjuntos como os valores em estações diferentes por exemplo, a análise do coeficiente de variação é interessante, ainda mais com outliers que podem afetar significativamente o seu valor.

$$CV = \frac{s}{\bar{x}}$$

11.3 Código em python do algoritmo de tratamento de outliers

Como o estudo do código foi dividido por modelo analisado e – ao longo do trabalho – várias versões e alternativas foram exploradas nos códigos, foram elaborados dois arquivos em python distintos, porém a lógica de ambos é a mesma, portanto será destacado o código para o Modelo de Perez porque ele possui linhas adicionais para a comparação de resultados com os valores do PVsyst. A diferença que será demonstrada é na abordagem inicial para aplicar o modelo de transposição, enquanto para o Modelo de Perez a irradiância total foi calculada a partir de uma única função que soma as 3 componentes (direta, difusa e refletida), no Modelo de Hay-Davies foram analisados os 3 gráficos individualmente e posteriormente os valores foram somados. Inclusive, ocorreram tentativas de replicar esse algoritmo para o Modelo de Perez para uma inspeção visual nos gráficos para verificar alguma possível incoerência nos valores de cada uma das componentes, porém a função equivalente para o cálculo somente do modelo de transposição de Perez, por algum motivo não identificado, não fornecia valores de irradiância na parte da manhã. Cogitou-se espelhar os valores da tarde para os horários matutinos, entretanto como isso causaria provavelmente uma distorção muito grande em relação aos valores reais a ideia foi descartada e a solução de usar uma única função que calcula a irradiância do plane of array com o Modelo de Perez foi utilizada.

O código foi executado no ambiente do Google Colab na versão gratuita que, atualmente, possui como CPU padrão o Intel Xeon com 2vCPU (CPU virtuais) e 13GB de memória RAM. É possível incrementar a quantidade de vCPU, a memória RAM e até mesmo utilizar uma GPU, porém o código foi executado no ambiente convencional, com as configurações acima e há algumas linhas no código para contabilizar o tempo necessário para sua execução, que será informado nos resultados mais adiante. Eles também foram executados no IDE do Noteable e as figuras abaixo, demonstrando trechos do código, são retiradas a partir do ambiente do Noteable para uma melhor visualização.

Além da biblioteca pvlib, primordial para o cálculo de irradiância do plane of array por conta de suas funções e a própria leitura do arquivo typical meteorological year (tmy) para que depois fosse possível manipular a matriz para extrair informações importantes para as equações. É necessário destacar as bibliotecas pandas e numpy foram muito importantes para manipulação dos vetores e listas e a matplotlib para plotar os diversos gráficos diferentes no trabalho. Por último e não menos importante a scipy que possibilitou, a partir da função curve_fit, a realização da regressão não-linear em si, além de fornecer ferramentas estatísticas e de análise de sinais fundamentais como as medidas de assimetria, curtose, coeficiente de variação e possibilitar realizar a transformada de Hilbert.

```
1 #Rafael Vilela Santa Rosa
2 !pip install -q pvlib
3 import time
4 import pvlib
5 from pvlib import location, irradiance, tools
6 from pvlib.iotools import read_tmy3
7 import pandas as pd
8 from matplotlib import pyplot as plt
9 import pathlib
10 import numpy as np
11 from scipy.optimize import curve_fit
12 from scipy.signal import hilbert
13 from scipy.stats import skew, kurtosis, variation
14
15 tempo_inicial=(time.time())
-
```

Figura 24 - Bibliotecas em python utilizadas para o código

Para começar o código há a extração das colunas de irradiação direta horizontal, direta normal e global horizontal. Em seguida é necessário, a partir da localidade que o arquivo meteorológico informa, definir o ângulo solar através da função get_solarposition. Outro componente importante para os dois modelos é a irradiação normal direta extraterrestre (acima da atmosfera, a irradiação direta normal do Sol) para ser utilizada dentro das funções de transposição. O ângulo ótimo do módulo fotovoltaico é baseado em um algoritmo desenvolvido na Universidade de Stanford (JACOBSON, 2018), sendo essa a fórmula para o hemisfério norte, como o valor ficou próximo de 30° esse foi o ângulo definido.

```

47 df = pd.DataFrame(index=times)
48 df['DHI'] = tmy['DHI'].values
49 df['DNI'] = tmy['DNI'].values
50 df['GHI'] = tmy['GHI'].values
51
52 max_value1 = df['DHI'].max()
53 print(f"Valor máximo DHI: {max_value1}")
54 max_value2 = df['DNI'].max()
55 print(f"Valor máximo DNI: {max_value2}")
56 max_value3 = df['GHI'].max()
57 print(f"Valor máximo GHI: {max_value3}")
58
59 solar_position = location.get_solarposition(times)
60
61 # calculate extra terrestrial radiation
62 dni_extra = pvlib.irradiance.get_extra_radiation(df.index)
63
64 print('/n')
65 Optimal_tilt_angle_NH = 1.3793 + Φ*(1.2011 + Φ*(-0.014404 + Φ*0.000080509))
66 print(f'Inclinação ótima:{Optimal_tilt_angle_NH}')

```

Figura 25 - Definições dos parâmetros

Enquanto no código para o Modelo de Hay foram utilizadas 3 funções, presentes no Apêndice B, para o Modelo de Perez foi utilizada somente a irradiance.get_total_irradiance com os parâmetros necessários para aplicar o modelo de transposição escolhido. O azimute foi definido como 90° sem nenhuma razão em específico, já que a intenção é que funcione com todos os azimutes possíveis (que são infinitos, porque há infinitos ângulos entre 0 e 360°, apesar de na prática somente os ângulos inteiros serem considerados). Além das irradiações definidas anteriormente, o zênite solar e o azimute solar são definidos a partir de funções do pvlib e a massa de ar relativa a partir da função get_relative_airmass, que precisa do zênite aparente da localidade para o seu cálculo. Após o novo vetor de plane of array "global" foi utilizada a função .fillna que preenche eventuais valores "not a number" por 0, o que apareceu em algumas ocasiões para valores que seriam 0 por tenderem ao menos ou mais infinito, deixando somente números no dataframe que será tratado mais adiante para a eliminação dos outliers.

```
70
71 df_poa = pvlib.irradiance.get_total_irradiance(
72                         surface_tilt=30, surface_azimuth=90,
73                         dhi=df['DHI'], dni=df['DNI'], ghi=df['GHI'],
74                         dni_extra=dni_extra,
75                         solar zenith=solar_position['apparent zenith'],
76                         solar azimuth=solar_position['azimuth'],
77                         airmass=pvlib.atmosphere.get relative airmass(solar_position['apparent zenith']),
78                         model='Perez')
79 print("\n-----Dados-----")
80
81 df_poa['poa_global'].fillna(0, inplace=True)
```

Figura 26 - Dataframe com valores de irradiação

4 métricas foram utilizadas a fim de comparar, além de somente com valores percentuais, o que será alterado na irradiação do plane of array após a regressão não-linear: a assimetria, o envelope das curvas diárias a partir da Transformada de Hilbert, a curtose e o coeficiente de variação (desvio padrão pela média). O laço for permite que ocorra a iteração ao longo do ano inteiro, sendo que as funções vão considerar os 24 valores de irradiação do dia em “day_data”, colocando os valores em listas até o último dia do ano. Após a saída do laço condicional a média é calculada.

```
111 skewness_list = []
112 envelope_list = []
113 kurtosis_list = []
114 cv_list = []
115 fs=1
116
117 # Loop por cada dia do ano
118 for day in pd.date_range('2023-01-01', '2023-12-31'):
119     # Get the data for the current day
120     day_data = df_poa.loc[day.strftime('%Y-%m-%d'), 'poa_global']
121
122     # Calculate Hilbert transform of the data
123     analytic_signal = hilbert(day_data)
124     amplitude_envelope = np.abs(analytic_signal)
125     instantaneous_phase = np.unwrap(np.angle(analytic_signal))
126     instantaneous_frequency = (np.diff.instantaneous_phase) / (2.0*np.pi) * fs
127
128     # Calculate skewness and append to list
129     skewness_list.append(skew(day_data))
130
131     # Calculate mean envelope and append to list
132     envelope_list.append(np.mean(amplitude_envelope))
133
134     # Calculate kurtosis and append to list
135     kurtosis_list.append(kurtosis(day_data))
136
137     # Calculate coefficient of variation and append to list
138     cv_list.append(variation(day_data))
139
140 mean_skewness = np.mean(skewness_list)
141 mean_envelope = np.mean(envelope_list)
142 mean_kurtosis = np.mean(kurtosis_list)
143 mean_cv = np.mean(cv_list)
144
145 print(f"Mean skewness: {mean_skewness}")
146 print(f"Mean envelope: {mean_envelope}")
147 print(f"Mean kurtosis: {mean_kurtosis}")
148 print(f"Mean Coefficient of Variation: {mean_cv}")
```

Figura 27 - Cálculo de algumas métricas relevantes

Antes da realização da regressão há a definição de uma função para definir uma senoide, a criação dos parâmetros (coeficientes) que serão utilizados pela função curve_fit e a modelagem para permitir um plot do gráfico de dispersão de irradiação ao longo do ano com a função de tendência (senoidal) sobreposta sobre ele.

```

326 # Função para representar a forma senoidal
327 def sinusoidal(x, a, b, c, d):
328     return a * np.sin(b * (x - np.radians(c))) + d
329
330 # Agregar os dados por dia
331 daily_irradiance = df_poa['poa_global'].resample('D').sum()
332
333 # Obter o dia do ano para usar como variável independente no ajuste
334 day_of_year = daily_irradiance.index.dayofyear
335
336 # Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
337 # O período é ajustado para 2*pi/365 para refletir o ciclo anual
338 # A fase horizontal é ajustada para colocar o pico no meio do ano (dia 182)
339 initial_parameters = [daily_irradiance.max(), 2 * np.pi / 365, 182, daily_irradiance.min()]
340
341 # Ajustar a função aos dados
342 parameters, _ = curve_fit(sinusoidal, day_of_year, daily_irradiance, p0=initial_parameters)
343
344 # Gerar um array de x para a função ajustada
345 x_fit = np.linspace(1, 365, 1000)
346
347 # Gerar os valores y da função ajustada
348 y_fit = sinusoidal(x_fit, *parameters)
349
350 print(f"A função senoidal de tendência é: {parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}")
351
352 # Plotar os dados originais e a função ajustada
353 plt.scatter(day_of_year, daily_irradiance, label='data')
354 plt.plot(x_fit, y_fit, color='red', label='fit')
355 plt.xlabel('Day of year')
356 plt.ylabel('Irradiação diária [Wh]')
357 plt.title('Sinusoidal fit to daily POA irradiance data - limitação de potência pelo percentil 98')
358 plt.legend()
359 plt.text(170, 0, f'{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}", ha='center', va='center')
360 plt.show()

```

Figura 28 - Modelagem da função de tendência

A regressão não-linear da função `curve_fit` exige alguns parâmetros para sua execução, sendo que a função escolhida para modelar os valores de irradiação diariamente foi a senoidal porque é uma das funções que melhor se assemelham ao comportamento na natureza da irradiação solar sobre uma determinada área na Terra. Antes de chamar a função faz-se necessário criar um dataframe com os coeficientes da função senoidal (definida antes no código) e outros parâmetros com os dias e um índice dos horários. É necessário antes de inicializar a função a estimativa de parâmetros iniciais com o ajuste do período para refletir o ciclo diário. Após a adequação, a função realiza o método de minimização dos quadrados através do algoritmo padrão dela, o Levenberg-Marquardt, retornando uma mensagem de falha e o dia relacionado caso ocorra uma. Funcionando sem nenhuma intercorrência os parâmetros de amplitude, período, deslocamento de fase e deslocamento vertical são armazenados.

```

410 ##### tratamento de outlier 2
411 # Create a new DataFrame to store the adjusted irradiance values
412 df_poa_adjusted = df_poa.copy()
413
414 daily_parameters = pd.DataFrame(columns=['a', 'b', 'c', 'd'],
415                                     index=df_poa.index.normalize().unique())
416
417 coef_upper = 1.15
418 coef_lower = 1.05
419
420 # Criação de um DataFrame para armazenar os parâmetros diários
421 daily_parameters = pd.DataFrame(index=pd.date_range('2023-01-01', '2023-12-31'),
422                                     columns=['amplitude', 'period', 'phase_shift',
423                                     'vertical_shift'])
424
425 # Loop por cada dia do ano
426 for day in pd.date_range('2023-01-01', '2023-12-31'):
427     # Obter os dados para o dia atual
428     day_data = df_poa.loc[day.strftime('%Y-%m-%d')]
429
430     # Obter a hora do dia para usar como variável independente no ajuste
431     hour_of_day = day_data.index.hour
432
433     # Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
434     # O período é ajustado para 2*pi para refletir o ciclo diário
435     initial_parameters = [day_data['poa_global'].max(), 2 * np.pi, 12,
436                           day_data['poa_global'].min()]
437
438     # Ajustar a função aos dados
439     try:
440         parameters, _ = curve_fit(sinusoidal, hour_of_day, day_data['poa_global'],
441                                     p0=initial_parameters, maxfev=5000)
442     except RuntimeError:
443         print(f"Failed to fit curve for day {day}")
444         continue
445
446     # Armazenar os parâmetros no DataFrame
447     daily_parameters.loc[day, ['amplitude', 'period', 'phase_shift', 'vertical_shift']] =
448     parameters

```

Figura 29 - Tratamento de outliers (Parte I)

Ao término do primeiro laço for o que foi realizado foi a regressão não-linear sobre os valores e a determinação dos parâmetros de uma função senoidal que melhor caracteriza o comportamento sobre cada um dos 365 dias. É realizado um outro loop for, diariamente, para ajustar os outliers a partir da função senoidal para cada um dos dias e os limites superiores e inferiores selecionados com a criação de 3 séries no pandas, uma para os limites (função de tendência vezes o coeficiente de limite) e outra para as funções senoidais. A partir de uma comparação com uma condicional if que determinar que "se o valor da irradiação para determinada hora i for superior ao limite superior da série o valor da irradiação será igual ao limite superior para aquela hora" ou se não (elif) "se o valor da irradiação para determinada hora i for inferior ao limite superior da série o valor da irradiação será igual ao limite inferior para aquela hora".

```

447 # Loop por cada dia do ano novamente para ajustar os outliers
448 for day in pd.date_range('2023-01-01', '2023-12-31'):
449     # Obter os dados para o dia atual
450     day_data = df.loc[day.strftime('%Y-%m-%d')]
451
452     # Obter a hora do dia para usar como variável independente no ajuste
453     hour_of_day = day_data.index.hour
454
455     # Obter os parâmetros para o dia atual
456     parameters = daily_parameters.loc[day]
457
458     # Calcular os valores da função de tendência para cada hora do dia
459     trend_values = sinusoidal(hour_of_day, *parameters)
460
461     # Definir um limite superior e inferior
462     upper_limit = trend_values * coef_upper
463     lower_limit = trend_values * coef_lower
464
465     # série pandas para os valores de tendência e limite superior/inferior para facilitar a
466     # manipulação
466     trend_series = pd.Series(trend_values, index=day_data.index)
467     upper_limit_series = pd.Series(upper_limit, index=day_data.index)
468     lower_limit_series = pd.Series(lower_limit, index=day_data.index)
469
470 for i in range(len(day_data)):
471     if day_data['total_irradiance'].iloc[i] > 0:
472         # Substituir valores nos dados diários
473         if day_data['total_irradiance'].iloc[i] > upper_limit_series.iloc[i]:
474             day_data['total_irradiance'].iloc[i] = upper_limit_series.iloc[i]
475         elif day_data['total_irradiance'].iloc[i] < lower_limit_series.iloc[i]:
476             day_data['total_irradiance'].iloc[i] = lower_limit_series.iloc[i]
477
478         # Armazenar os dados ajustados de volta no DataFrame original
479         df.loc[day.strftime('%Y-%m-%d'), 'total_irradiance'] = day_data['total_irradiance']
480
481 # df['total_irradiance'] ajustado para outliers com base na tendência senoidal diária
482
483 # para valores negativos por conta do ajuste da senoide
484 df['total_irradiance'] = df['total_irradiance'].clip(lower=0)
485
486 print("\n-----Termino de tratamento de outlier-----")

```

Figura 30 - Tratamento de outliers (parte II)

Após alguns gráficos sobre as características das curvas de irradiação e de irradiação anual, são preparados os vetores de erros para realizar a comparação dos novos valores ajustados através do tratamento de outliers e os valores originais após aplicar as equações e modelos para definir o total de irradiação em um módulo fotovoltaico. Lembrando que os dataframes “df1” e “df2” não são dos valores de irradiação, portanto a comparação é realizada entre os valores originais utilizando o modelo de transposição de Perez para cada um dos softwares, mostrando através de barras de erros e percentuais de diferença no gráfico como que os valores mudaram em comparação com os valores simulados em software.

```

606 months = list(range(1, 13))
607 errors1 = np.abs(df3['percentual'] - 100)
608 errors2 = np.abs(df4['percentual'] - 100)
609 errors3 = np.abs(df5['percentual'] - 100)
610 errors4 = np.abs(df6['percentual'] - 100)
611
612
613 # Plot HelioScope data
614 plt.errorbar(months, df3['percentual'], yerr=errors1, fmt='o', color='blue',
   label='Original (em relação ao HelioScope)')
615
616 plt.errorbar(months, df5['percentual'], yerr=errors3, fmt='o', color='red', label='Com
   correção de outlier (em relação ao HelioScope)')
617
618 # linha em y=100
619 plt.axhline(100, color='black', linestyle='--')
620
621 plt.title('Comparação de percentuais mensais (Perez/HelioScope)')
622 plt.xlabel('Mês')
623 plt.ylabel('Percentual (%)')
624
625 plt.legend()
626
627 for month, error1, error3 in zip(months, errors1, errors3):
628     plt.text(month, df3['percentual'][month-1], f'{error1:.1f}%', ha='center', va='bottom')
629     plt.text(month, df5['percentual'][month-1], f'{error3:.1f}%', ha='center', va='top')
630
631 plt.legend()
632
633 # Exibir o gráfico
634 plt.show()

```

Figura 31 - Comparação entre os valores com e sem correção de outliers

Na parte final do código há novamente um novo levantamento de medidas estatísticas e de análises de sinais para verificar qual foi o efeito da regressão não-linear sobre algumas características das curvas de irradiação. Apesar de utilizarem o mesmo dataframe original para comparar não há problema, visto que anteriormente foi criado um dataframe “cópia” para tratar os outliers e depois ele foi copiado para o dataframe original, a única diferença foram os nomes de listas diferentes para posteriormente (porque não foi realizado nesse trabalho) poder trabalhar mais a fundo nas medidas de assimetria, envelope da curva, curtose e do coeficiente de variação para poder comparar de forma semelhante aos gráficos com diferenças percentuais acima. Não obstante, no final do código há uma variável para definir o tempo de término do código e depois mostrar em tela o tempo computacional para todo o algoritmo, que é relevante ao considerar a abordagem da regressão horária, visto que – conforme detalhado mais a fundo na próxima seção e na conclusão – tornou-se significativo.

```

663 skewness_list2 = []
664 envelope_list2 = []
665 kurtosis_list2 = []
666 cv_list2 = []
667
668 # Loop por cada dia do ano
669 for day in pd.date_range('2023-01-01', '2023-12-31'):
670     # Get the data for the current day
671     day_data = df_poa.loc[day.strftime('%Y-%m-%d'), 'poa_global']
672
673     # Calculate Hilbert transform of the data
674     analytic_signal = hilbert(day_data)
675     amplitude_envelope = np.abs(analytic_signal)
676     instantaneous_phase = np.unwrap(np.angle(analytic_signal))
677     instantaneous_frequency = (np.diff.instantaneous_phase) / (2.0*np.pi) * fs
678
679     # Calculate skewness and append to list
680     skewness_list2.append(skew(day_data))
681
682     # Calculate mean envelope and append to list
683     envelope_list2.append(np.mean(amplitude_envelope))
684
685     # Calculate kurtosis and append to list
686     kurtosis_list2.append(kurtosis(day_data))
687
688     # Calculate coefficient of variation and append to list
689     cv_list2.append(variation(day_data))
690
691 mean_skewness = np.mean(skewness_list2)
692 mean_envelope = np.mean(envelope_list2)
693 mean_kurtosis = np.mean(kurtosis_list2)
694 mean_cv = np.mean(cv_list2)
695
696 print(f"Mean skewness: {mean_skewness}")
697 print(f"Mean envelope: {mean_envelope}")
698 print(f"Mean kurtosis: {mean_kurtosis}")
699 print(f"Mean Coefficient of Variation: {mean_cv}")
700
701 tempo_final=(time.time())
702 tempo=tempo_final - tempo_inicial
703
704 print(f"{tempo} segundos")
705 print("Fim!")

```

Figura 32 - Análise de métricas relevantes após o tratamento de outliers

12. Resultados

Ao executar o código na mesma base meteorológica do Aeroporto Internacional de Piedmont Triad, que é possível importar diretamente das funções no pvlib (no arquivo '723170TYA.CSV'), algumas análises o próprio código realiza determinadas ações dependendo dos valores, entretanto algumas decisões e valores nos códigos foram selecionados a partir dos gráficos resultantes. Como será exposto com mais detalhes na seção de conclusão, há melhorias para automatizar ainda mais o código, algumas nas próprias funções, porém da forma que está escrita a quantidade de valores que o usuário necessita alterar é bem reduzida.

São dois códigos quase semelhantes, com a diferença que para a base meteorológica utilizada determinadas ações tornaram-se necessárias para um modelo e outras não, estando comentadas para não afetar o tratamento de outliers, além do fato do Modelo de Perez ser executado dentro de uma função que calcula toda irradiância no plano, enquanto no Modelo de Hay-Davies as 3 funções foram executadas individualmente e em um segundo momento a irradiância das 3 somadas para um mesmo dataframe do pandas para o tratamento dos outliers.

Antes de explicar os resultados, é importante frisar que o tempo de execução dos códigos no Google Colab foi semelhante, por volta de 40s na primeira execução com a importação das bibliotecas e de 30s caso ele seja executado com as bibliotecas já importadas, sendo a seção onde é realizado o laço for iterando diariamente as curvas de tendência e substituindo os valores acima ou abaixo dos limites definidos a mais demorada de todas, quase metade do tempo de execução dos códigos. Na IDE do Noteable, apesar de não deixar explícito se a CPU é equivalente ao do Google Colab, na sua versão gratuita com 1 vCPU e 4GB de memória RAM o tempo computacional foi semelhante.

12.1 Resultados do algoritmo no Modelo de Perez

De imediato necessitou-se uma abordagem diferente do que o inicialmente proposto, que era somente aplicar a função de tendência resultante de regressão não-linear devido aos altos valores de irradiância em determinadas horas. Lembrando que, apesar da constante solar possuir um valor aproximadamente de 1361W/m², a irradiância em determinada área pode ser superior por conta da irradiância difusa e refletida pelas superfícies próximas, porém é muito raro acontecer.

A partir da constatação anterior decidiu-se utilizar determinado percentil para limitar os valores máximos de irradiância, a partir da análise manual o percentil 98 (P98) foi o escolhido, com um valor de cerca de 1013W/m², mais coerente do que o percentil 99 que possui o valor de aproximadamente 1651W/m². O percentil 97, de valor aproximado de 856W/m² também foi considerado, porém pela quantidade de valores acima do nonagésimo oitavo foram elevados optou-se pelo valor próximo aos STC. Para fins de comparação, o terceiro quartil é de somente 157.78W/m².

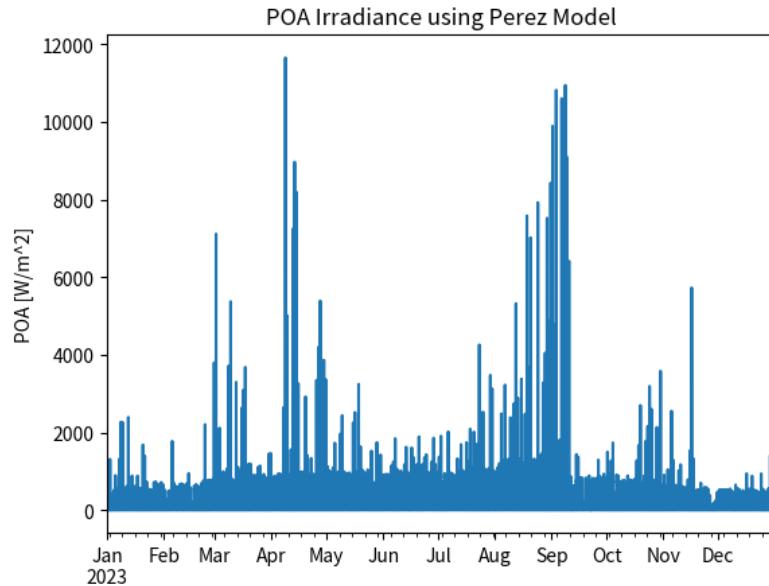


Figura 33 - Irradiância no plane of array ao longo do ano sem tratamento de outliers

Assim como será utilizado no código para o modelo de Hay-Davies, o mês de Junho será utilizado para visualização da irradiância mensal, a fim de avaliar a coerência das curvas. Como é possível observar, exceto por alguns outliers as curvas possuem uma coerência, levando em conta que a base meteorológica se situa no hemisfério norte e – evidentemente – a potência que os módulos fotovoltaicos estão sujeitos é superior no verão que é no meio do ano nesse hemisfério.

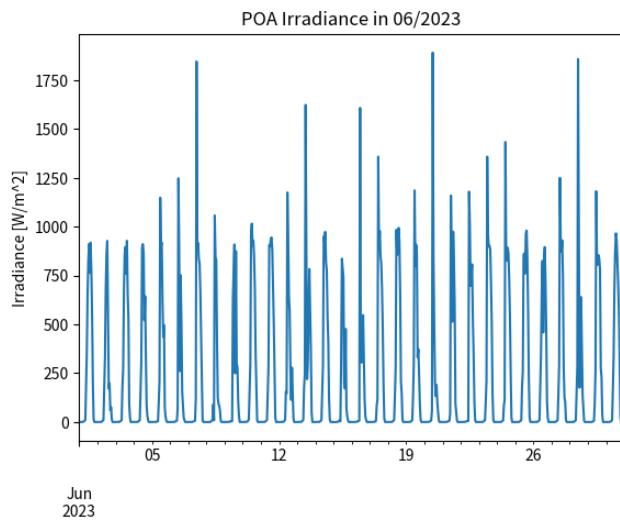


Figura 34 - Irradiância no plane of array em Junho sem tratamento de outliers

É necessário ter atenção com o gráfico abaixo que no eixo das ordenadas (y) o valor é da irradiação (energia) e não da irradiância; o motivo é computacional e pelo fato de ser mais fácil de visualizar a dispersão porque no eixo das abscissas (x) há 365 pontos ao invés de mais de 8700 pontos nos outros gráficos que são horários. É muito claro a discrepância absurda entre determinados valores de mais de 15kWh/dia de energia enquanto a curva é muito inferior. Os valores muito abaixo da curva, pela baixa ocorrência, são provavelmente devido a condições específicas do tempo em determinados dias, recorrendo outras análises em mais valores do dataset escolhido, entretanto não ocorreu devido a confiabilidade dele. Os valores altos são, principalmente, devido aos modelos de transposição utilizados para o módulo fotovoltaico analisado, que possui inclinação de 30° e azimute de 90° (porém em algumas funções foi considerado o ângulo de 270° que – considerando as condições ideais e desprezando um

efeito térmico significativo sobre o módulo – deve possuir um valor próximo de irradiação ao longo do ano.

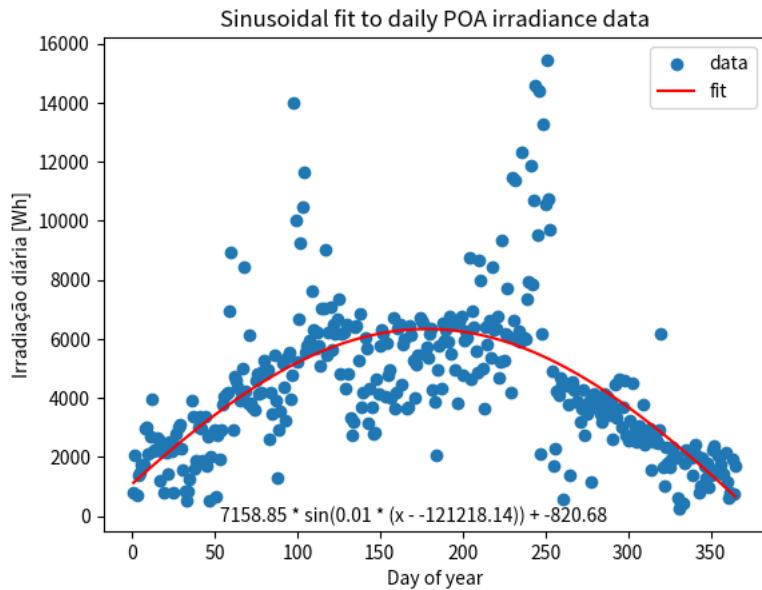


Figura 35 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária

Como mencionado anteriormente, ocorreu uma limitação de potência com o limite sendo o percentil 98 que não afeta somente os dias com um valor de energia excessivamente alto já que em alguns dias de baixa irradiação ocorreram os outliers, por isso outros pares ordenados (data) no gráfico deslocaram-se, desconsiderando a distorção pela nova escala vertical do gráfico.

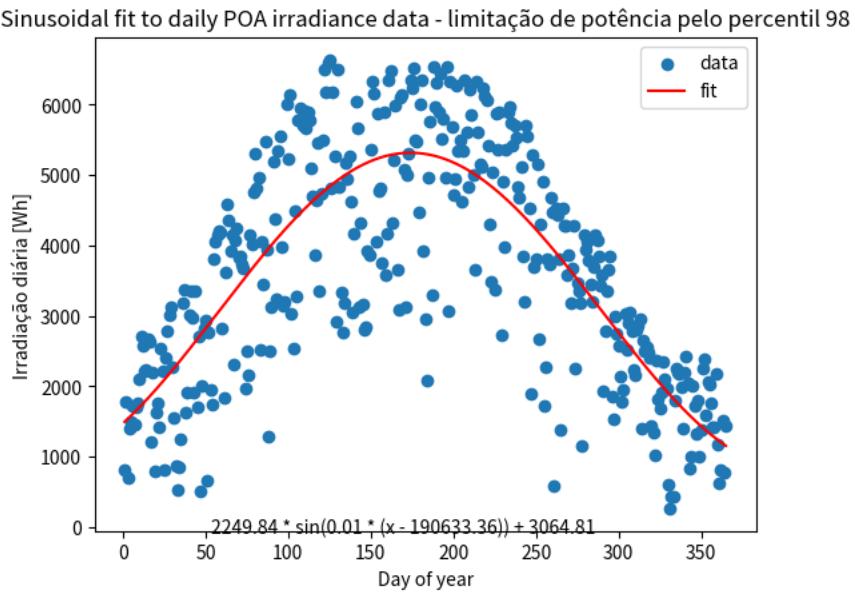


Figura 36 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária - com limitação de potência pelo percentil 98

Após esse primeiro tratamento de outliers foi realizado a regressão não-linear em cada dia do ano, resultando no seguinte gráfico de irradiação ao longo do ano. Após a regressão poucos valores ficaram limitados pela condição anterior e a curva ficou mais coerente do que é esperado para um módulo fotovoltaico com uma inclinação próxima da ideal, de acordo com o algoritmo escolhido, sem sombreamento e desconsiderando outras perdas inerentes, como as perdas internas do módulo e sujeira externa nas camadas acima das células fotovoltaicas.

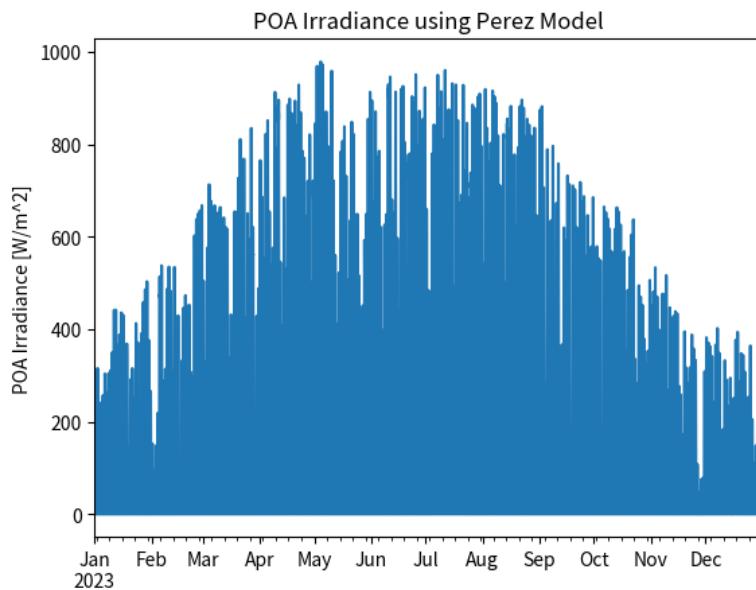


Figura 37 - Irradiância no plane of array ao longo do ano com tratamento de outliers

O gráfico do mês de Junho sofreu algumas mudanças, em especial a limitação dos valores de irradiância discrepantes com valores razoáveis, além do dia 7 de Junho onde ocorreu uma gráfica em "W", porém, por não ser uma anormalidade que impactaria o resultado final, não ocorreu um estudo aprofundado sobre o comportamento deste dia em específico. De forma geral, o mês de Junho e outros meses permaneceram sem grandes distorções, sem contar que para o foco do trabalho, não há importância na metamorfose dos valores de irradiância, somente nos valores de irradiação, seja a nível diário, mensal ou anual.

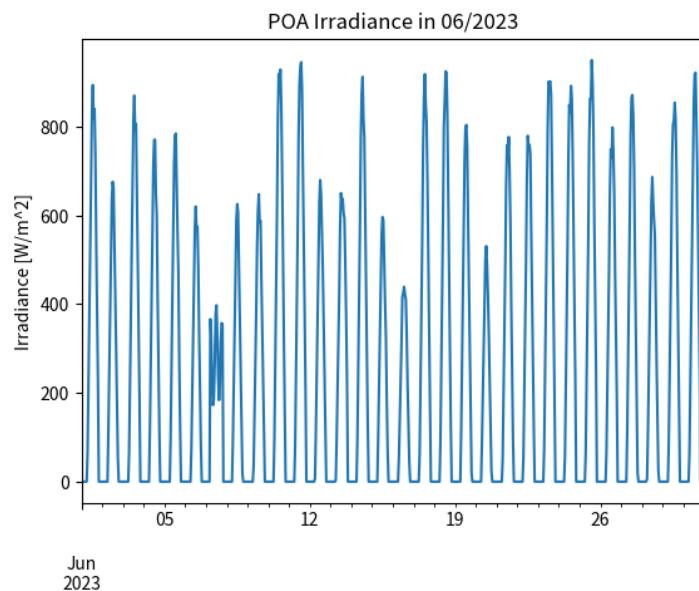


Figura 38 - Irradiância no plane of array em Junho com tratamento de outliers

São centenas de funções senoidais diferentes para modelar a irradiância diária, entretanto o deslocamento de fase em relação ao meio-dia resultou em valores positivos de madrugada, o que é incorreto. Para contornar esse problema de potência de madrugada a simples condicional para comparar o valor de irradiância original do modelo e negar a modelagem pela curva de tendência para valores iguais a 0 W/m² foi suficiente. Outro fato que ocorreu foi, pior ainda, o surgimento de valores negativos

de potência! Para evitá-lo a solução foi tão elegante quanto o pré-processamento da curva antes da regressão, a função .clip da biblioteca pandas foi utilizada para todo valor de irradiância negativo, transformando-os em 0.

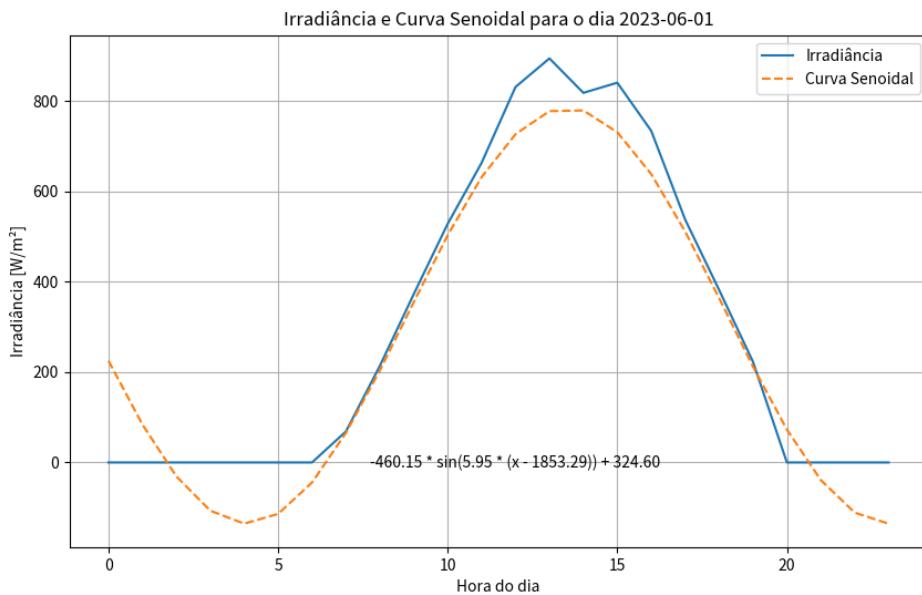


Figura 39 - Regressão não-linear senoidal das curvas diárias de irradiância

A partir da regressão não-linear, para os coeficientes de limite superior e inferior de 1.15 e 1.05, respectivamente, o gráfico de dispersão com a (nova) função de tendência abaixo foi elaborado. É interessante notar que em determinadas áreas ocorreu uma concentração maior de pontos próximos a função de tendência. Como, para substituir os outliers, ocorreu uma substituição a nível diário não é possível observar assíntotas polinomiais nos valores de energia, entretanto está comentado nos dois códigos o “tratamento de outlier 1”, que era baseado no ajuste da energia diária, transformando a irradiância constante ao longo dos dias, mas esta solução a nível diário foi escolhida por conseguir melhor ajustar os meses, mesmo os com maior número de outliers.

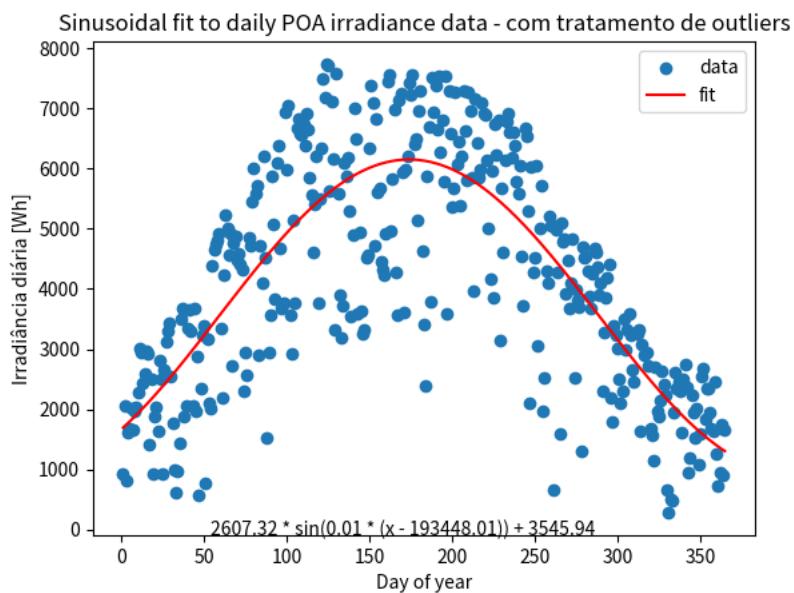


Figura 40 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária - com tratamento de outliers

A partir dos valores de energia diário foi computado o valor mensal de energia mensal, posteriormente verificado o valor percentual em relação aos softwares de simulação HelioScope e PVsyst, simulados com o modelo de transposição de Perez e os resultados ficaram mais próximos do que o modelo sem o tratamento de outliers. Em valores: a energia anual da modelagem com a limitação pelo percentil e sem a regressão foi equivalente a 88,99% e 82,45% em relação ao HelioScope e PVsyst, respectivamente. Após a modelagem os percentuais alteraram-se para 102,20% e 94,71%.

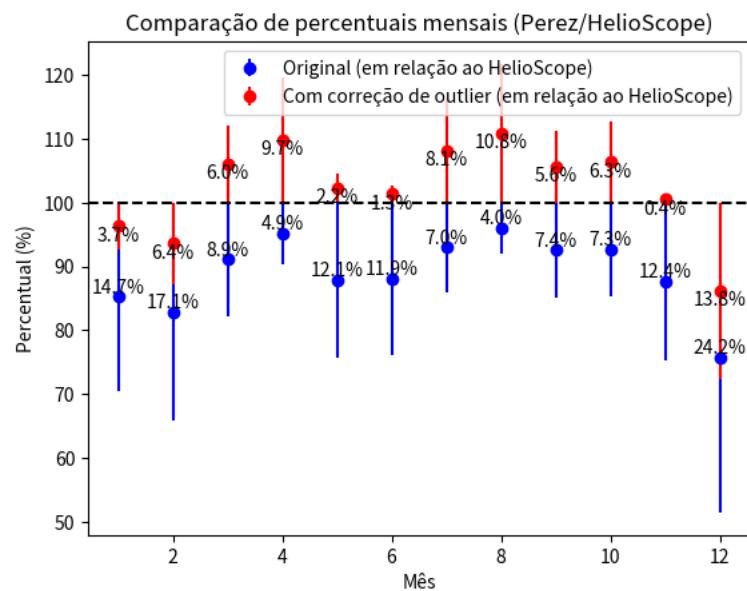


Figura 41 - Comparação dos valores mensais de energia antes e após a regressão - Perez/HelioScope

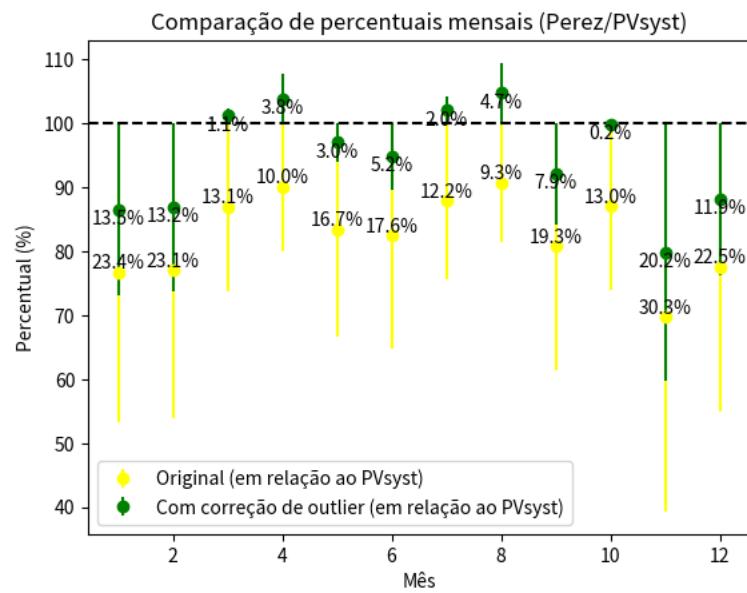


Figura 42 - Comparação dos valores mensais de energia antes e após a regressão - Perez/PVsyst

Projeto de Graduação

Tabela 8 - POA simulado no HelioScope (Perez)

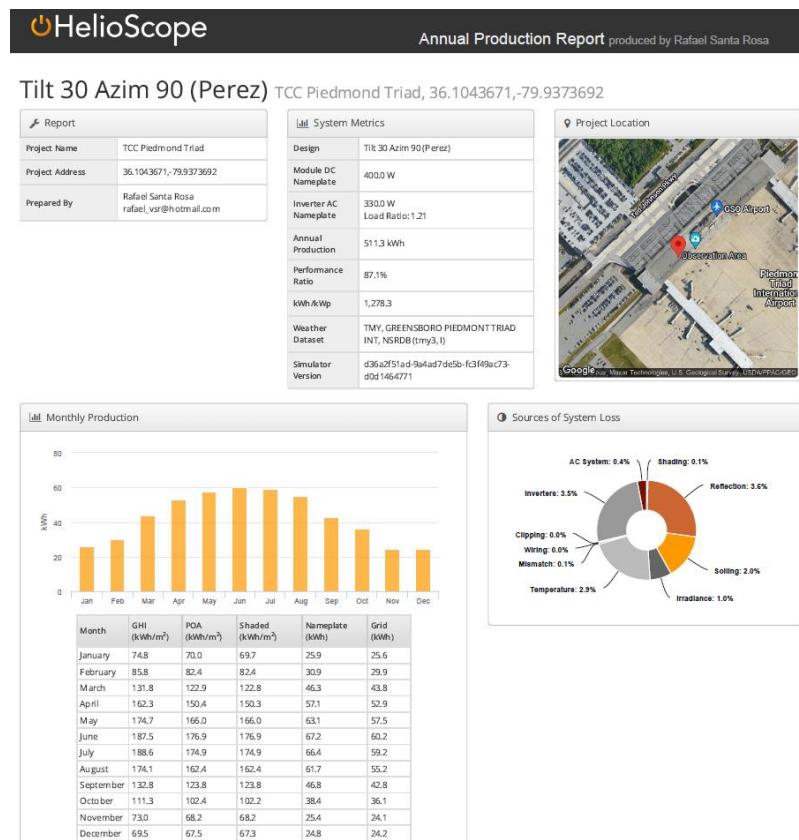
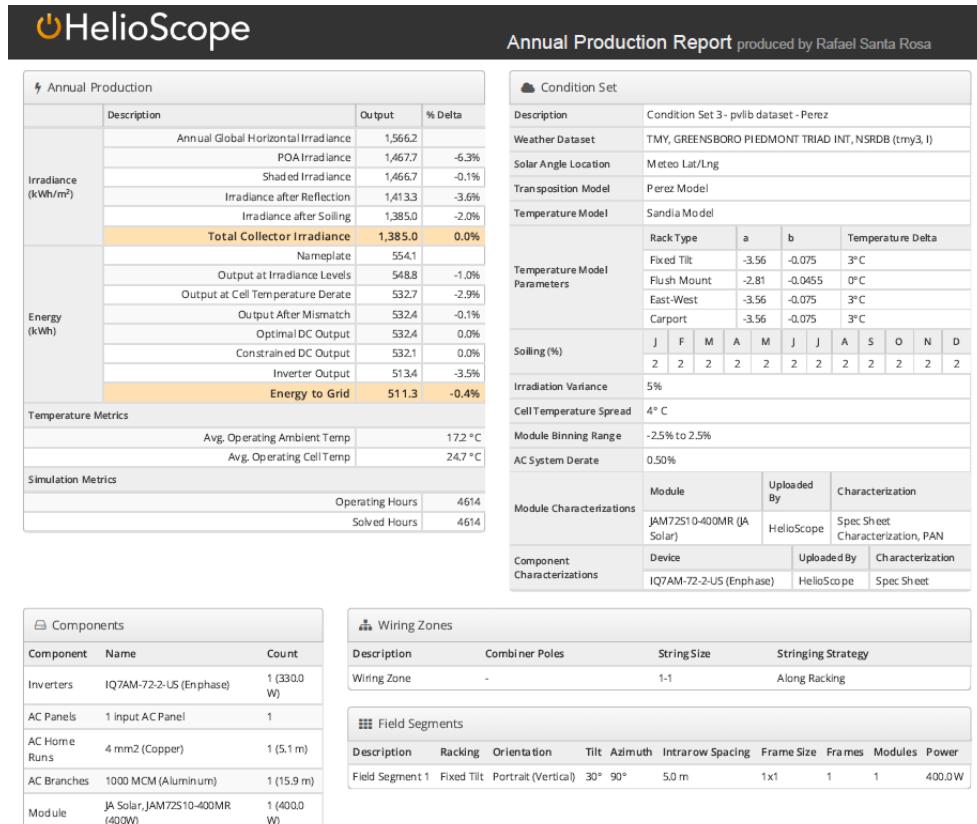
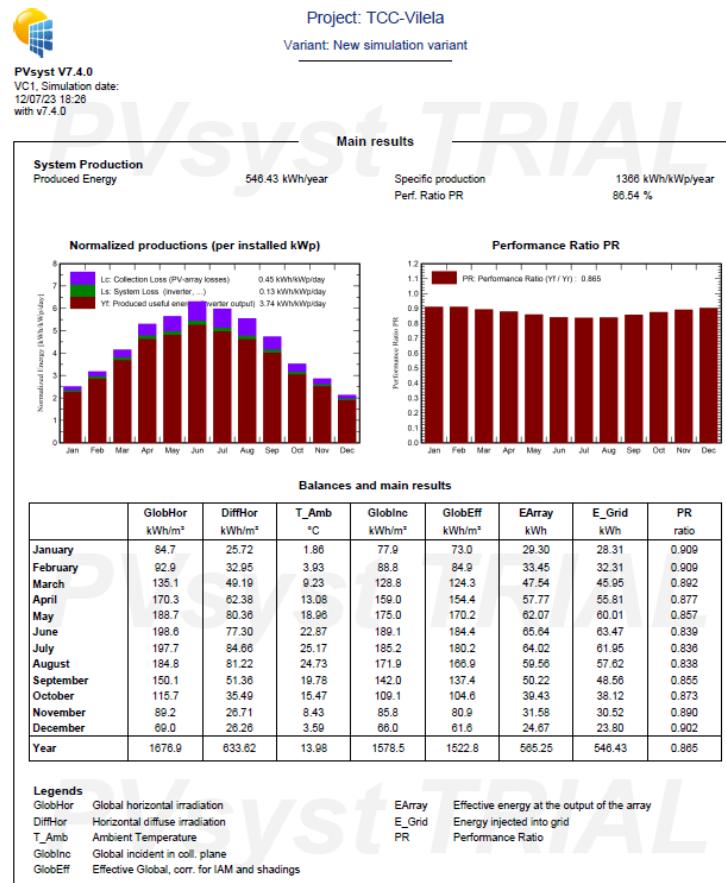


Tabela 9 - Condições de simulação HelioScope (Perez)



Projeto de Graduação

Tabela 10 - POA simulado no PVsyst (Perez)



Quadro 1 - Condições de simulação PVsyst (Perez)

Project: TCC-Vilela
Variant: New simulation variant

PVsyst V7.4.0
VC1, Simulation date:
12/07/23 18:26
with v7.4.0

General parameters									
Grid-Connected System					Sheds on ground				
PV Field Orientation					Sheds configuration				
Orientation									
Fixed plane									
Tilt/Azimuth	30 / -90 °								
Horizon					Near Shadings				
Free Horizon					Linear shadings				
					User's needs				
					Unlimited load (grid)				
					Perez				
					Imported separate				

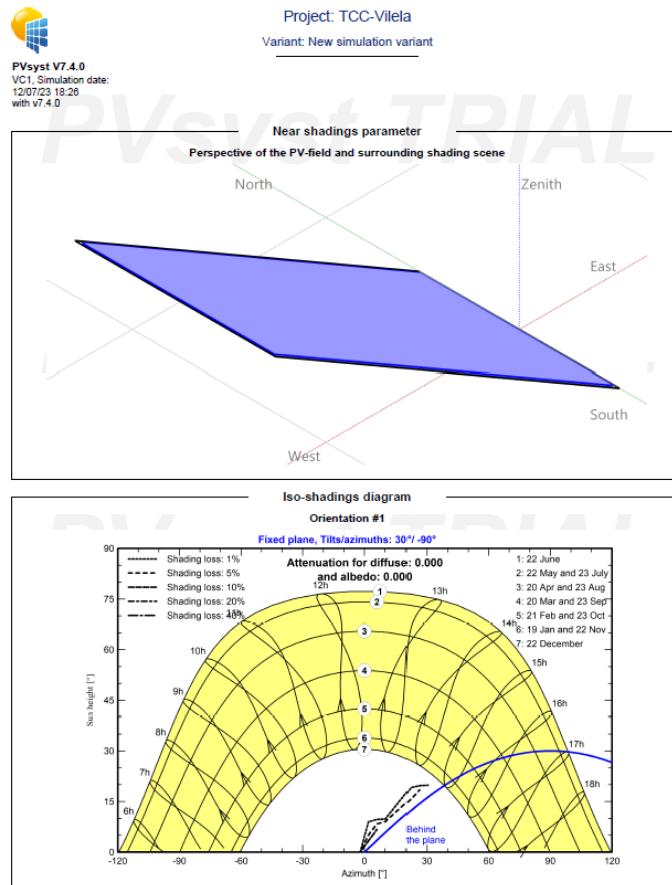


Figura 43 - Condições de simulação do PVsyst (Perez)

12.2 Resultados do algoritmo no Modelo de Hay-Davies

Para o Modelo de Hay a abordagem foi igual a inicialmente proposta, que era somente aplicar a função de tendência resultante de regressão não-linear. Os maiores valores de irradiância foram inferiores a 1000W/m^2 , plausíveis considerando todas as componentes que determinam a irradiância sobre uma superfície inclinada.

A partir da constatação anterior nenhum percentil foi utilizado para limitar os valores máximos de irradiância, porém foram verificados para constatar a distribuição a partir, principalmente, dos valores dos quartis. Mesmo o gráfico assemelhando-se e muito ao valor do gráfico do plane of array do Modelo de Perez ele foi submetido ao mesmo processo de regressão não-linear a partir de uma função senoidal, limitação por um limite superior e outro inferior de forma manual e comparado com os valores de energia do Modelo de Hay-Davies sem o tratamento de outliers.

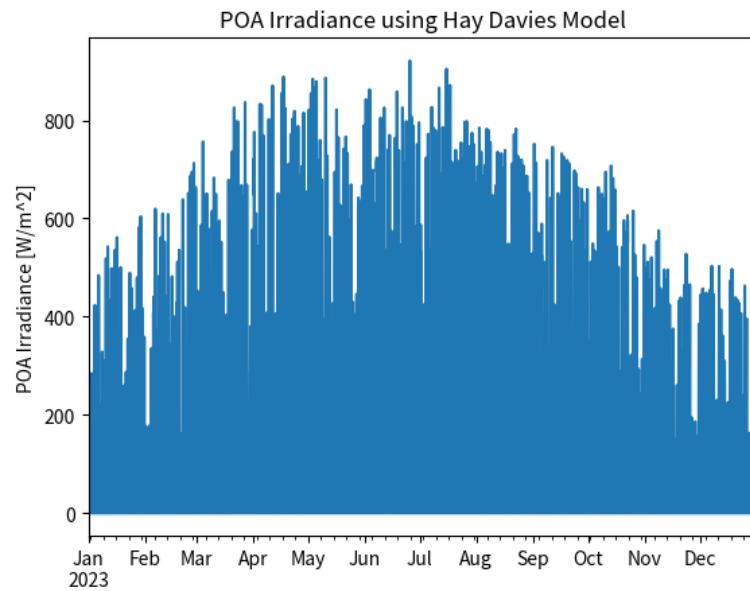


Figura 44 - Irradiância no plane of array ao longo do ano sem tratamento de outliers

Assim como no código para o modelo de Perez, o mês de Junho será utilizado para visualização da irradiância mensal, a fim de avaliar a coerência das curvas. Como é possível observar, a percepção inicial é que não há outliers significativos a partir do modelo de Hay-Davies, porém é necessária uma aplicação da matemática para poder extrair com uma maior certeza se é necessário ou não aplicar algum algoritmo para ajustar os valores de irradiância nesta situação.

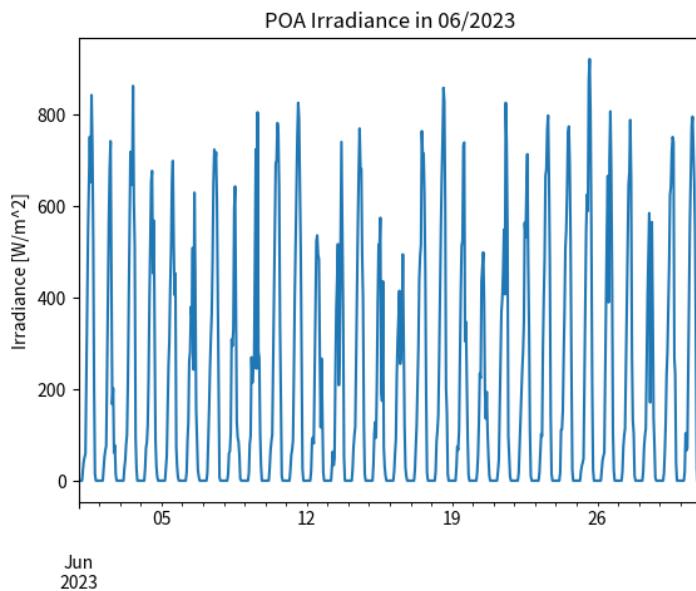


Figura 45 - Irradiância no plane of array em Junho sem tratamento de outliers

Há uma certa evidência que uma discrepância entre determinados valores existir, devido a dispersão dos pontos ser permearem uma área significativa, enquanto espera-se uma área menor pela aglutinação dos valores ao redor da função de tendência. Os valores muito abaixo da curva, pela baixa ocorrência, são provavelmente devido a condições específicas do tempo em determinados dias, conforme dissertado nos resultados do Modelo de Perez. Há alguns valores de irradiação bem superior ao esperado entre o segundo e terceiro trimestre do ano; considerando a hipótese do código relacionado ao modelo do anterior, a aplicação do algoritmo possui uma justificativa para ser realizada, a fim de ajustar os valores dos outliers presentes.

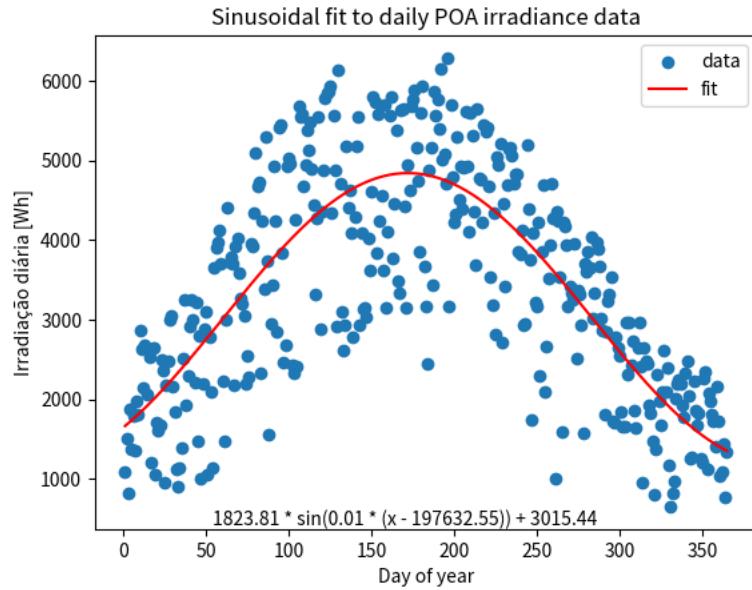


Figura 46 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária

Após a regressão não-linear em cada dia do ano, resultando no seguinte gráfico de irradiação ao longo do ano, visualmente não parece que ocorreu uma melhoria em si. A curva, pelo menos, ficou mais coerente do que é esperado para um módulo fotovoltaico com uma inclinação próxima da curva de potência de um módulo fotovoltaico em condições parcialmente ideais, portanto há uma esperança de que no final o resultado fique melhor do que a simples aplicação do modelo de transposição, de irradiação refletida e direta no código.

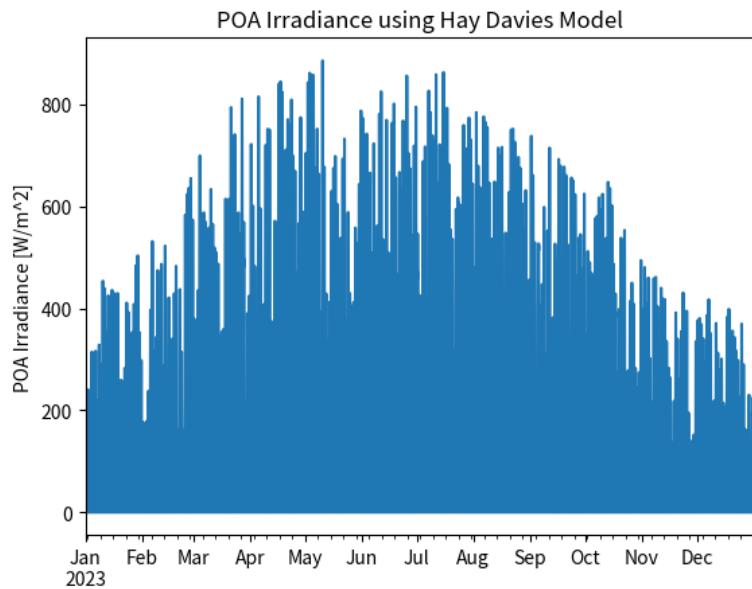


Figura 47 - Comparaçao dos valores mensais de energia antes e aps a regresso - Hay-Davies/HelioScope

O gráfico do mês de Junho não sofreu mudanças visuais significativas. De forma geral, o mês de Junho e outros meses permaneceram sem grandes distorções, assim como no Modelo de Perez pós-regressão. Outro fato é que o dia 7 de Junho não ficou “distorcido” como no outro gráfico anterior, novamente, não sendo um foco de preocupação aquele fato.

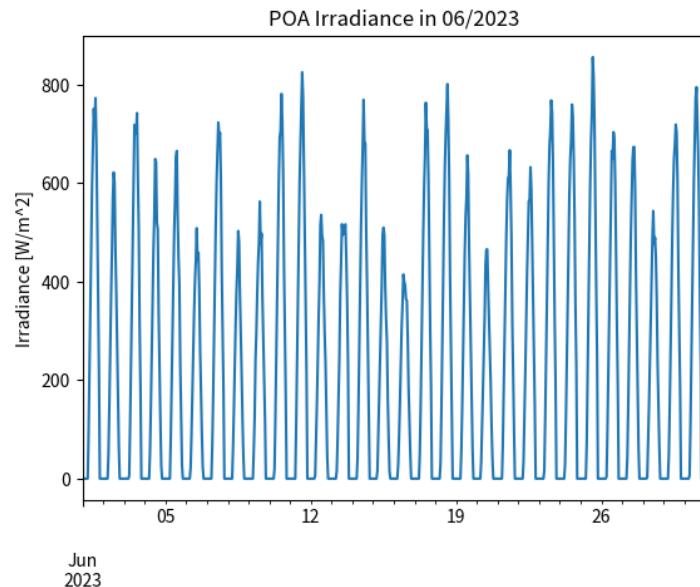


Figura 48 - Irradiância no plane of array em Junho com tratamento de outliers

A elegante função senoidal é extremamente semelhante a anterior exposta no Modelo de Perez, mas com valores de amplitude, fase e frequência distintos.

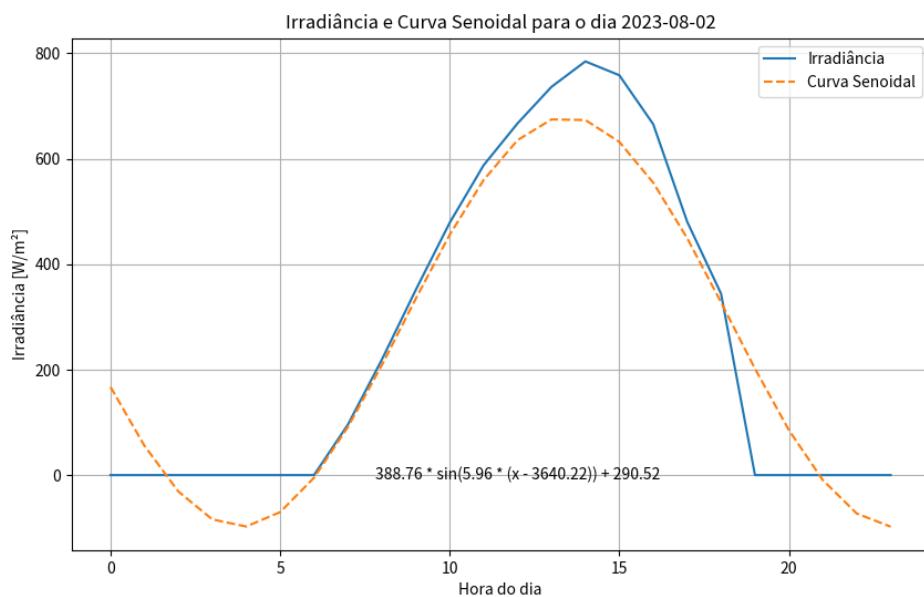


Figura 49 - Regressão não-linear senoidal das curvas diárias de irradiância

A partir da regressão não-linear o gráfico de dispersão com a (nova) função de tendência abaixo foi elaborado. Da mesma forma que ocorreu no outro modelo, uma maior concentração dos valores ocorreu ao redor da função de tendência. Exceto por um valor no meio do ano que foi bem inferior, em termos de energia, do valor da curva de tendência, a área de dispersão do “polígono irregular” diminuiu. Em suma, visualmente, há indicações de que a limitação de outliers deve resultar em uma maior precisão dos valores em relação ao parâmetro do HelioScope, já que não foi possível selecionar no PVsyst o Modelo de Hay-Davies.

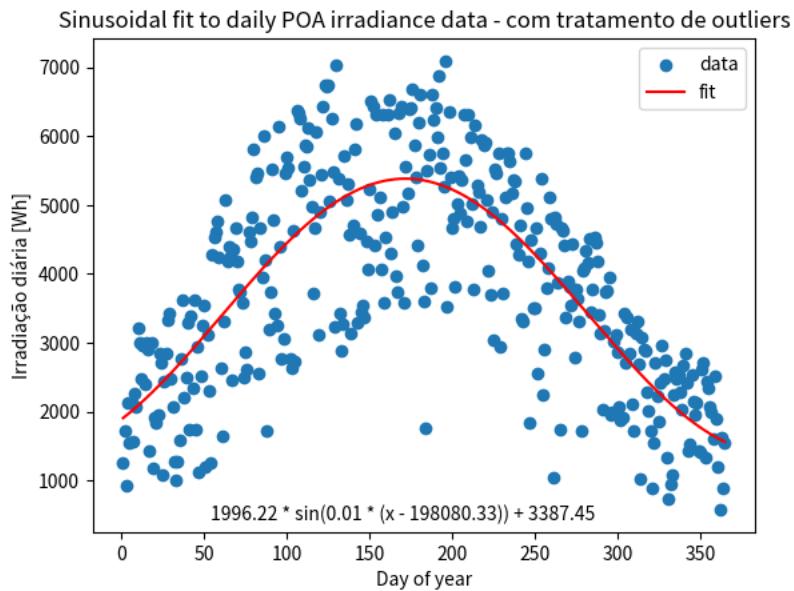


Figura 50 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária - com tratamento de outliers

A partir dos valores de energia diário foi computado o valor mensal de energia mensal, posteriormente verificado o valor percentual em relação ao softwares de simulação HelioScope, simulado com o modelo de transposição de Hay-Davies e os resultados – como já esperado – ficaram mais próximos do que o modelo sem o tratamento de outliers. Em valores: a energia anual da modelagem sem a regressão foi equivalente a 84,73% em relação ao HelioScope, após a modelagem o percentual foi alterado para 94,98%. A regressão desse código conseguiu aproximar muito 4 valores dos valores simulados em software (menos de 2% de diferença percentual), enquanto os demais não ficaram tão distantes em relação a não aplicação da regressão não-linear.

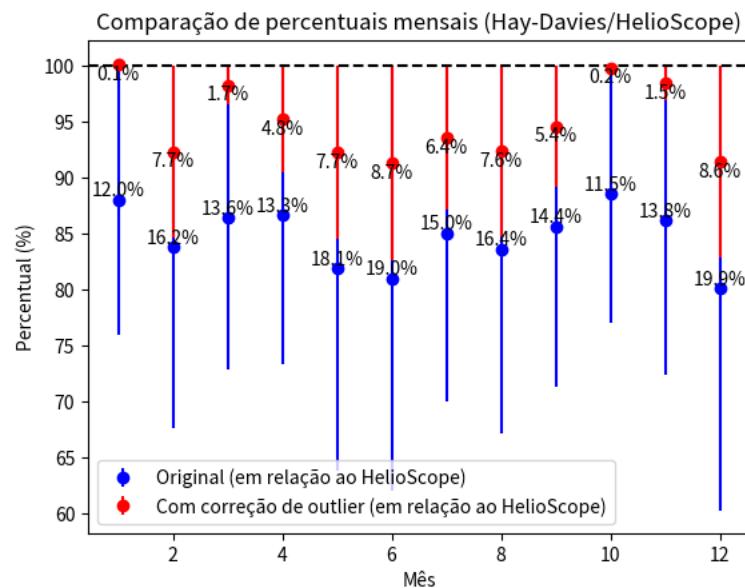


Figura 51 - Comparação dos valores mensais de energia antes e após a regressão - Hay-Davies/HelioScope

Projeto de Graduação

Tabela 11 - POA simulado no HelioScope (Hay)

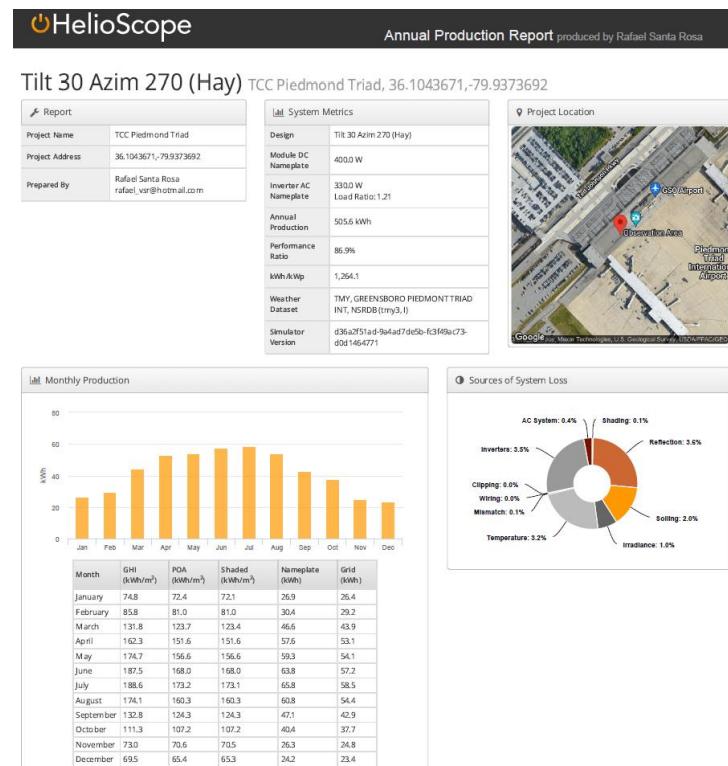


Tabela 12 - Condições de simulação HelioScope (Hay)

Annual Production		Condition Set							
Irradiance (kWh/m²)	Description	Output	% Delta						
	Annual Global Horizontal Irradiance	1,566.2							
	POA Irradiance	1,454.3	-7.1%						
	Shaded Irradiance	1,453.4	-0.1%						
	Irradiance after Reflection	1,401.0	-3.6%						
	Irradiance after Soiling	1,373.0	-2.0%						
	Total Collector Irradiance	1,373.0	0.0%						
Energy (kWh)	Nameplate	549.3							
	Output at Irradiance Levels	544.0	-1.0%						
	Output at Cell Temperature Derate	526.6	-3.2%						
	Output After Mismatch	526.3	-0.1%						
	Optimal DC Output	526.3	0.0%						
	Constrained DC Output	526.2	0.0%						
	Inverter Output	507.7	-3.5%						
	Energy to Grid	505.6	-0.4%						
Temperature Metrics	Avg. Operating Ambient Temp	17.2 °C							
	Avg. Operating Cell Temp	24.6 °C							
Simulation Metrics	Operating Hours	4614							
	Solved Hours	4614							
Components									
Component	Name	Count							
Inverters	IQ7AM-72-2-US (Enphase)	1 (330.0 W)							
AC Panels	1 input AC Panel	1							
AC Home Runs	4 mm2 (Copper)	1 (6.4 m)							
AC Branches	1000 MCM (Aluminum)	1 (7.0 m)							
Module	JASolar JAM7Z510-400MR (400W)	1 (400.0 W)							
Wiring Zones									
Description	Combiner Poles	String Size	Stringing Strategy						
Wiring Zone	-	1-1	Along Racking						
Field Segments									
Description	Racking	Orientation	Tilt	Azimuth	Intrarow Spacing	Frame Size	Frames	Modules	Power
Field Segment 1	Fixed Tilt	Portrait (Vertical)	30°	270°	5.0 m	1x1	1	1	400.0 W

12.3 Resultados das medidas estatísticas e de análise de sinal

Após a demonstração dos resultados percentuais dos códigos para ambos os modelos de transposição, os últimos valores a serem analisados são as medidas estatísticas médias de assimetria, curtose e coeficiente de variação e o envelope através da Transformada de Hilbert calculados primeiro para os dois modelos sem o modelo de transposição e depois do tratamento dos outliers.

Tabela 13 - Características das curvas de irradiância antes do tratamento de outliers

Cenário	Assimetria média	Curtose média	CV médio	Envelope médio (W/m ²)
Com o Modelo de Perez	2,33	5,66	212 %	376,94
Com o Modelo de Hay-Davies	1,07	-0,30	137 %	238,81

Após o tratamento de outliers a assimetria média para os dois cenários diminuiu, dessa forma aproximando as curvas de irradiância a uma distribuição normal, o coeficiente de variação diminuiu (como verificado visualmente na menor dispersão dos valores ao redor da função de tendência) e a curtose média indica que a curva de irradiância do Modelo de Perez está com uma função de probabilidade mais "achatada" que a distribuição de normal, tornando-a platicúrtica igual à do Modelo de Hay-Davies, em média, inclusive com esta última acentuando essa característica por aumentar o seu valor negativo. O envelope médio do Modelo de Perez diminuiu e o do Modelo de Hay-Davies aumentou um pouco, indicando que o primeiro, em média, diminuiu a magnitude de suas irradiancias diárias e o segundo, apesar de visualmente não ser identificável, aumentou ligeiramente em relação ao que era antes.

Tabela 14 - Características das curvas de irradiância após o tratamento de outliers

Cenário	Assimetria média	Curtose média	CV médio	Envelope médio (W/m ²)
Com o Modelo de Perez	0,75	-1,06	125 %	283,69
Com o Modelo de Hay-Davies	0,64	-1,19	118 %	251,62

Conforme descrito no começo, o escopo do trabalho é prover um método de aplicar um algoritmo de regressão não-linear – devido à natureza da fenômeno da irradiação solar - amplamente aplicado em diversas áreas diferentes, de uma forma que pode ser utilizado de uma forma mais ampla para diversas situações. Apesar de ser uma versão piloto ou alfa, por assim dizer, os resultados alcançados foram bastante significativos, sendo o mais importante o destaque em relação a concepção da correção de valores dispersos que serão destacados adiante.

13. Conclusão

Conforme descrito no começo, o escopo do trabalho é prover um método de aplicar um algoritmo de regressão não-linear – devido a natureza da fenômeno da irradiação solar - amplamente aplicado em diversas áreas diferentes, de uma forma que pode ser utilizado de uma forma mais ampla para diversas situações. Apesar de ser uma versão piloto ou alfa, por assim dizer, os resultados alcançados foram bastante significativos, sendo o mais importante o destaque em relação a concepção da correção de valores dispersos que serão destacados adiante.

Em relação aos valores brutos, a diferença percentual da energia anual e dos meses para a irradiação utilizando o Modelo de Perez e o modelo de Hay-Davies diminuiu, tanto utilizando o software HelioScope quanto o software PVsyst como parâmetros “reais”. Importante ressaltar que ambos habilitam a utilização dos modelos de transposição e levam em consideração basicamente a mesma abordagem matemática para calcular os valores, para cada tipo de condição de inclinação do módulo fotovoltaico, azimute, equação para a massa de ar relativa e outros parâmetros haverá um peso diferente do modelo de transposição sobre o valor total de irradiação sobre o plano inclinado (módulo fotovoltaico monofacial e em alguma estrutura fixa). O intuito desse algoritmo não foi ser utilizado para comparar com os resultados do plane of array de um software necessariamente, pode-se compará-lo (ou utilizá-lo) com dados meteorológicos de satélite por exemplo, visto que em última análise – por mais que os processos de obtenção de valores de irradiação sejam completamente diferentes – o comportamento da irradiação seguirá uma tendência aproximadamente senoidal.

Outros valores que possibilitam um melhor entendimento da regressão não-linear com o algoritmo de Levenberg-Marquardt podem ter sua implementação futura no código como as medidas de assimetria e curvose; de antemão a justificativa primordial de sua utilização é a ampla aceitação de sua eficácia tanto que é a versão default da função `curve_fit` da `scipy.optimize` da biblioteca `scipy` em python. Os outros dois algoritmos são o Trust Region Reflective e o “Dogleg” que é um tipo de algoritmo baseado no método de região de confiação (trust region), entretanto os dois possuem um custo computacional mais elevado, de acordo com os próprios desenvolvedores da biblioteca, portanto – considerando que é um código piloto – o algoritmo escolhido foi o de Levenberg-Marquardt que combina o método de Gauss-Newton e o do gradiente descendente para convergir com rapidez e robustez para resolução dos problemas de mínimos quadrados.

Há diversas possibilidades de melhoria que estão disponíveis de imediato ao código, principalmente para automatizá-lo e não ter a necessidade imediata de colocar o valor manualmente de alguns parâmetros como o limite superior e inferior da curva de tendência. A aplicação ou não de uma assíntota horizontal para valores muito discrepantes de irradiação pode ser determinada a partir de uma constante máxima, não necessariamente a constante solar, como por exemplo um determinado percentil do conjunto de valores de irradiação. Não foi realizado nesse trabalho porque seria necessário um maior entendimento de conceitos de estatística, análise de sinais no domínio do tempo e da frequência e processamento de dados e de um estudo mais aprofundado em si dos fenômenos de irradiação solar, precisão de equipamentos de metrologia, sendo que provavelmente a melhor solução seria a aplicação de um treinamento de máquina para determinar os valores a partir de um grande conjunto de dados fornecidos de fontes meteorológicas confiáveis. O Modelo de Perez em si, por exemplo, foi testado e elaborado a partir de 13 locais diferentes dos Estados Unidos e Europa, com bases de 6 meses a 3 anos dependendo do local. Mesmo sendo elaborado em 1990 ainda é o principal modelo de transposição para definir a irradiação do céu difuso, um dos 3 componentes que compõe a irradiação para um plena inclinado (plane of array), apesar de existir alguns ajustes e conceitos aprimorados a base é a mesma que foi formulada há 3 décadas, mostrando sua robustez!

O código elaborado foi para a situação mais simples de instalação de módulo fotovoltaico porque os módulos que estão fixados em solar trackers ou seguidores solares possuem uma incidência totalmente diferente, visto que todas 3 componentes são afetadas pela mudança de azimute (e por vezes do ângulo), além das fórmulas para o cálculo de outras componentes importantes como o zênite e azimute solares não ser o mesmo. Uma complicação que pode surgir, dependendo da concepção do seguidor, é a sombra (sendo penumbra ou umbra) que os módulos ou fileiras podem ocasionar nos demais. Como dito anteriormente, o código foi otimizado ao considerar um cenário onde os módulos fotovoltaicos não iriam ser afetados significativamente por sombras, apesar de, caso a sombra seja a mesma ao longo do ano inteiro e os valores meteorológicos a considerarem (por exemplo, algum projeto que colocará os

instrumentos já considerando a sombra) provavelmente a regressão não-linear conseguirá surtir um efeito semelhante por conta do padrão cíclico que a sombra fará.

Outra condição em que o código não deverá ser tão efetivo é em presença de módulos fotovoltaicos bifaciais, desde que os mesmos, evidentemente, estejam de tal forma que haverá um ganho bifacial devido a superfície traseira, em um terreno de albedo moderado a alto e em determinada geometria que favoreça que a irradiância difusa e refletida (eventualmente alguma irradiância direta normal). Para o lado frontal deverá funcionar da mesma forma, visto que é igual a módulos monofaciais de mesma tecnologia, número de células, topologia do arranjo das células e outras condições, entretanto, globalmente, a potência de saída do módulo dependerá de uma estimativa coerente da face traseira. Uma alternativa seria conjugar os valores de irradiância difusa e refletida principalmente com piranômetros e outros instrumentos no local específico onde os módulos fotovoltaicos bifaciais estão ou ficarão.

Quadro 2 - Estações meteorológicas utilizadas no Modelo de Perez (PEREZ, 1990)

Site	Climate/Environment Main Features	Data Set Span and Frequency
Geneva, Switzerland [42]	Temperate maritime, with central Europe continental influence. Persistent nebulosity enhanced by "blocking position at foot-hill of the Alps."	1 yr. hourly data
Trappes, France [43]	Temperate maritime with high incidence of intermediate skies	3 yr. hourly data
Carpentras, France [43]	Mediterranean	3 yr. hourly data
Albany, NY, USA [44,45]	Humid continental with bimodal	3yr. hourly data
New York, NY, USA [45]	Humid continental with maritime influence plus large City's anthropogenic environment	2 yr. 15 min. data
Farmingdale, NY, USA [45]	Same as above but without city's environment	1 yr. 15 min. data
Oswego, NY, USA [45]	Humid continental, Great Lakes basin	6 mo. 15 min. data
Glens Falls, NY, USA [45]	Humid continental	6 mo. 15 min. data
Phoenix, AZ, USA [46]	Arid, low elevation	6 mo. hourly data
Albuquerque, NM, USA [46]	Arid, High elevation (1800 m)	1 yr. hourly data
Los Angeles, CA, USA [46]	Arid and maritime influence plus high frequency of anthropogenic smog events	6 mo. hourly data
Osage, KS, USA [46]	Continental, U.S. Great Plains	6 mo. hourly data
C. Canaveral, FL, USA [46]	Subtropical, low latitude, maritime	6 mo. hourly data

Em síntese, nos 3 cenários investigados e com os resultados que foram demonstrados antes da conclusão é possível dizer que o algoritmo presente nos códigos – que foram separados em dois, porém possuem a mesma concepção – foi efetivo e pode ser utilizado para uma estimação de irradiância que incide sobre um módulo fotovoltaico ou até mesmo para outras aplicações como para a agricultura em plantações de terrenos inclinados. Enquanto para um cenário simples a validação necessita de menos parâmetros de entrada, a fim de aprimorá-lo para a utilização em outros cenários novas mudanças seriam necessárias, inclusive a utilização de alguma outra função além da senoidal para modelagem da função de tendência por alguma determinada particularidade encontrada.

A abordagem da simulação do código a partir de uma simulação no software SAM também é recomendada, não foi realizada por conta de valores divergentes de simulação para essa localidade em específico, com valores do plane of array muito divergentes dos valores encontrados nos outros dois softwares. De qualquer forma, a validação do código a partir de outros programas, além do SAM, como o PV*SOL (ou PVSOL) seria fundamental para verificar sua versatilidade entre os demais softwares de simulação existentes no mercado.

Quanto a eficiência computacional, é possível aprimorar o código ao reformular a quantidade de variáveis globais e locais, melhorias nas funções e principalmente nos loops condicionais. Como o foco do trabalho não era almejar o menor custo computacional possível o código foi escrito de forma a ficar mais clara para entender como o algoritmo de regressão não-linear deveria ser concebido. Certos gráficos poderiam ser evitados, porém não foram comentados para uma análise visual do comportamento dos valores de irradiação e irradiância ao longo do processo, para entender melhor quais pontos poderiam causar incoerências como a modelagem da função seno provando gerando valores negativos de irradiância caso certas condições estivessem omissas no código. Para um conjunto de dados horários o código possuiu um desempenho razoável, em menos de 1 minuto conseguiu ser executado por completo em configurações de um computador convencional, porém para um conjunto de dados a cada 30 minutos ou 15 minutos já seria interessante otimizar o código para um tempo de execução menor.

Quadro 3 - Conjuntos meteorológicos da NREL (NREL, s.d.)

Region	Model Name	Satellite	Temporal Resolution	Spatial Resolution	Years Covered
Europe, Africa, & Asia	PSM V3	METEOSAT IODC	15, 30, 60-minute	4km	2017-2019
USA & Americas	PSM V3	GOES	30, 60-minute	4km	1998-2021
USA & Americas	PSM V3	GOES	10, 30, 60-minute	4km	2019-2021
USA (Continental) & Mexico	PSM V3	GOES	5, 30, 60-minute	2km	2019-2021
South Asia	SUNY	METEOSAT IODC	60-minute	10km	2000-2014
Asia, Australia & Pacific	PSM V3	Himawari	10, 30, 60-minute	2km	2016-2020
Asia, Australia & Pacific	PSM V3	Himawari	30, 60-minute	4km	2011-2015

Com certeza há muitos detalhes que podem aprimorar o código de tal forma que possa ser utilizado em diferentes situações e cenários mais extremos para modelar as curvas de irradiância diária, sendo que no código há comentado a primeira hipótese levantada, porém depois deixada em segundo plano que era a de transformar as curvas de irradiância em retas (constantes) diárias, diminuindo a área de dispersão dos valores ao longo do ano.

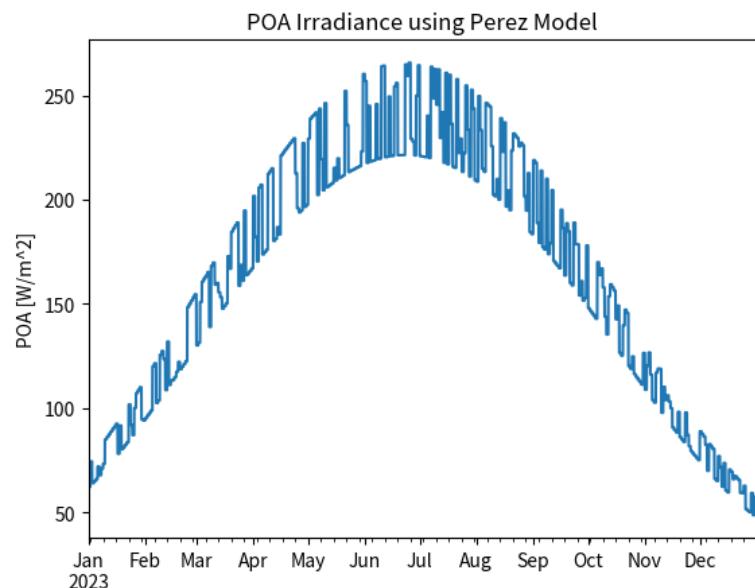


Figura 52 - Irradiância no plane of array ao longo do ano com tratamento de outliers pela energia diária

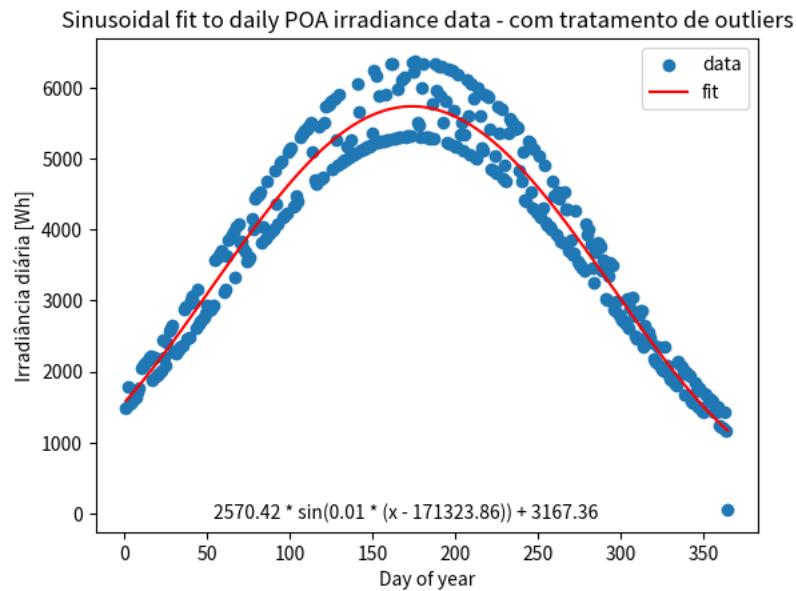


Figura 53 - Curva de tendência senoidal ao longo do ano no gráfico de dispersão de energia diária

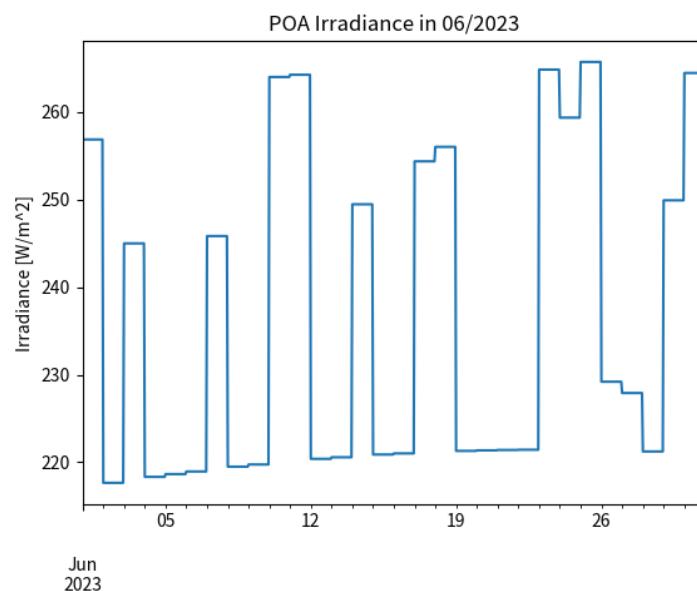


Figura 54 - Irradiância no plane of array em Junho com tratamento de outliers pela energia diária

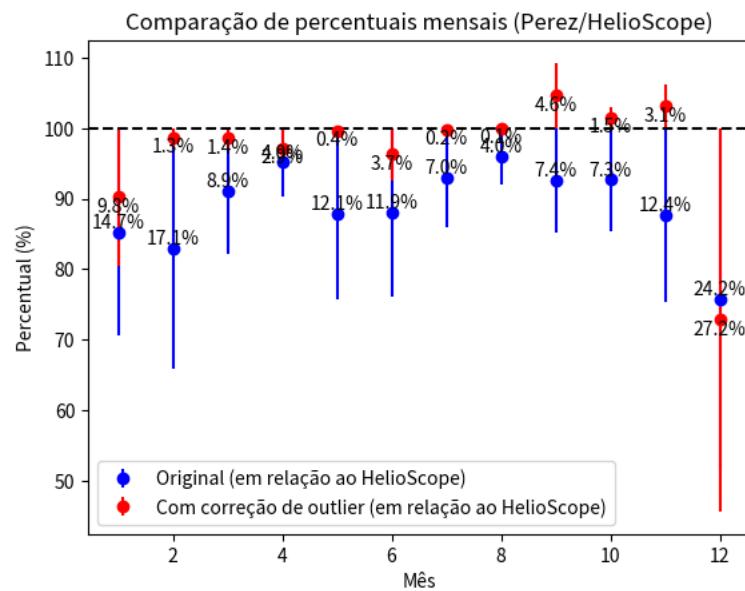


Figura 55 - Comparação dos valores mensais de energia antes e após a regressão - Perez/HelioScope

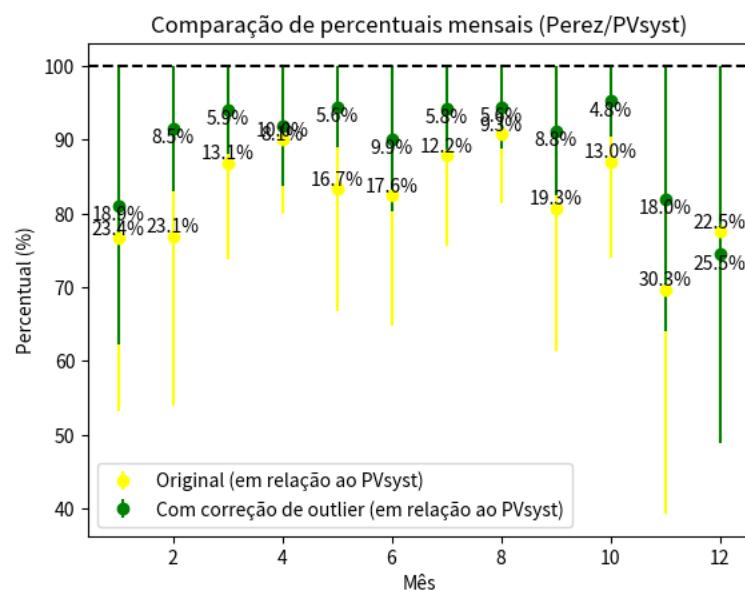


Figura 56 - Comparação dos valores mensais de energia antes e após a regressão - Perez/PVsyst

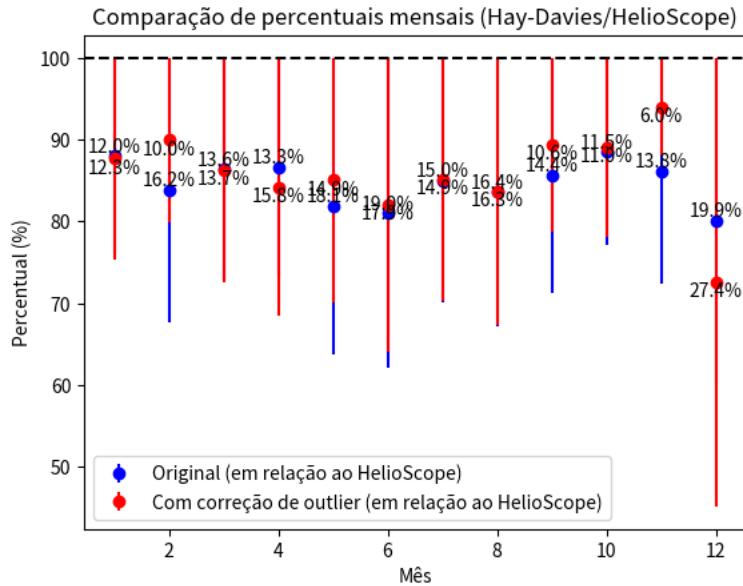


Figura 57 - Comparação dos valores mensais de energia antes e após a regressão - Hay-Davies/HelioScope

Conforme visualizado nas figuras acima, com valores limites semelhantes aos utilizados para o tratamento de outlier considerado na seção de resultados, a curva de energia para esse tratamento possui duas assíntotas senoidais limitando a potência a valores positivos próximos a função de tendência, ao longo de todas as horas diárias em um ano inteiro. Como a intenção é modelar os valores para estimar a energia em um período maior como o mensal ou anual essa outra abordagem deve estar nas alternativas para comparar com a proposta neste trabalho, porém considerando os mesmos limites superiores e inferiores o erro percentual foi maior para o mesmo cenário analisado na seção de resultados acima. Para o modelo de Hay-Davies inclusive o resultado foi pior ainda, sendo que na maior parte do ano os resultados ficaram praticamente os mesmos.

Uma possível alternativa seria trabalhar com os dois modelos de tratamento propostos no código, com o modelo de regressão não-linear que ajusta a energia diária ao redor de uma assíntota sendo utilizada como um referencial para que outra função de tendência seja utilizada em dias em que, por mais que a curva de irradiância com valores horários seja ajustada, a irradiação no dia ainda seja muito divergente da irradiação esperada para o dia. Como esta condição é observável em determinados pontos do gráfico de dispersão de irradiação para ambos os modelos de transposição, um ajuste diferente pode ser necessário nesses dias, não necessariamente alterando bruscamente os valores de irradiância porque realmente alguns dias extremamente nublados no verão ou irradiância acima do normal no inverno inevitavelmente existirão, por fenômenos meteorológicos pouco comuns ou raros, conforme citados na seção de meteorologia.

Um adendo ao tratamento de outlier original, pelo menos para a localidade e base meteorológica escolhida, os pontos do gráfico de dispersão não ficaram tão concentrados ao redor da curva de tendência quanto ao outro código proposto, portanto é importante um aprofundamento no estudo entre a escolha entre minimizar a área de dispersão ou aumentar a concentração de valores ao redor de uma função. O intuito do trabalho não foi de realizar essa análise, entretanto há possíveis campos de investigação entre as duas diferentes abordagens existentes, considerando a natureza da irradiação solar para uma aplicação de um módulo fotovoltaico inclinado.

Novas simulações utilizando localidades com longitudes e latitudes diferentes, de preferência em regiões com uma grande concentração de bases meteorológicas próximas podem ajudar a aprimorar o modelo ao comparar as medidas estatísticas e de análise de sinais presentes no código e – como mencionado anteriormente – outras medidas que não foram utilizadas no estudo acima. Caso o algoritmo não seja efetivo em condições mais extremas como em regiões de alta latitude ou com climas que possuam grande variação de temperatura, por exemplo, uma alternativa seria testar os outros dois algoritmos que a própria função `curve_fit` disponibiliza. Como mencionado ainda nas conclusões, uma conjugação de diferentes algoritmos de regressão, com condições específicas de assimetria ou coeficiente de variação diária da curva de irradiância determinando qual algoritmo deve ser utilizado para determinado dia, estão entre as hipóteses levantadas para melhoria do código.

Tabela 15 - Comparação anual final entre os cenários (valor do software dividido pelo valor simulado)

Cenário em relação ao simulado nos softwares	Modelo de Perez - HelioScope	Modelo de Perez - PVsyst	Modelo de Hay-Davies - HelioScope
Sem tratamento de outliers	Não considerado	Não considerado	84,73 %
Com limitação de irradiação máxima	88,99 %	82,45 %	Não foi utilizado
Com tratamento de outliers e limitação de irradiação máxima	102,20 %	94,71%	94,98 %

Portanto, o êxito deste trabalho em si não foi o resultado ser positivo e melhor que a abordagem inicial ("tratamento 1 no código em python") e sim a validação de formas de melhorar a modelagem matemática da irradiação diária e da irradiação mensal e anual para módulos fotovoltaicos monofaciais e fixos, considerando os principais modelos de transposição aplicados atualmente e de outras modelos de irradiação necessários para estimar a irradiação em um plano inclinado.

14. Referências

ABSOLAR. **Panorama da solar fotovoltaica no Brasil e no mundo.** 2023. Disponível em: <https://www.absolar.org.br/mercado/infografico/>. Acesso em: 25 jun 2023.

BLAIR, N; DIORIO, N; FREEMAN, J; GILMAN, P; JANZOU, S; NEISES, T; WAGNER, M. **System Advisor Model (SAM) General Description (Version 2017.9.5).** 2018. Disponível em <https://www.nrel.gov/docs/fy18osti/70414.pdf> Acesso em: 9 jun 2023.

BOYLE, G. **Renewable Energy: Power for a Sustainable Future.** 2nd ed. Oxford, UK: Oxford University Press, 2004.

BREHM, N; BAYLISS, A; CHRISTL, M; SYNAL, H; CZYMZIK, M. **Eleven-year Solar Cycles Over the Last Millennium Revealed by Cosmogenic Radionuclides.** Solar Physics, 2021. 296 p.

DUFFIE, J; BECKMAN, W. **Solar engineering of thermal processes.** John Wiley & Sons. 2013.

EINSTEIN, A. **Generation and transformation of light.** Annalen der Physik, vol. 17, 1905.

ELVINA, Mac. **Modeling 101.** 2023. Disponível em: <https://help-center.helioscope.com/hc/en-us/articles/13804165205395-Modeling-101>. Acesso em: 16 jun 2023.

ENCYCLOPÆDIA BRITANNICA. **Electromagnetic radiation | Spectrum, Examples, & Types.** In Encyclopædia Britannica. (n.d.). Disponível em: <https://www.britannica.com/science/electromagnetic-radiation>. Acesso em: 29 abr 2023.

ENCYCLOPÆDIA BRITANNICA. **Photon | Definition, Discovery, Charge, & Facts.** In Encyclopædia Britannica. (n.d.). Disponível em: <https://www.britannica.com/science/photon>. Acesso em: 29 abr 2023.

FEYNMAN, R. **QED : The Strange Theory of Light and Matter.** Princeton University Press. 1985.

GRAY, D; LEE, H; SUTTER-FELLA, C; YANG, W; GUEYMARD, C. **Performance modeling of 20% efficient solar cells.** Proceedings of the 37th IEEE Photovoltaic Specialists Conference (PVSC), 001174-001179. 2011. doi: 10.1109/PVSC.2011.6185772.

GREEN, M et al. **Solar Cell Efficiency Tables (Version 54).** Progress in Photovoltaics: Research and Applications, 2018. 26(1), 3-12 p.

HARROUNI, S. **Modeling Solar Radiation at the Earth's Surface.** Proceedings of the 37th IEEE Photovoltaic Specialists Conference (PVSC), 001174-001179. 2008. doi: 10.1109/PVSC.2011.6185772.

HAY, J; DAVIES, J. **Calculations of the solar radiation incident on an inclined surface.** In: Hay, J.E., Won, T.K. (Eds.), Proc. of First Canadian Solar Radiation Data Workshop, 59. Ministry of Supply and Services, Canada. 1980.

HELIOSCOPE. **HelioScope: Mathematical Formulation.** 2013. Disponível em: <https://s3.amazonaws.com/helpscout.net/docs/assets/5889260f2c7d3a7846304e89/attachments/589528a4dd8c8e73b3e94e17/HelioScope---Mathematical-Formulation-2013-03-28.pdf>. Acesso em: 10 jun 2023.

HERSCHEL, W. **On the Thermal Spectrum of the Sun's Rays.** In Philosophical Transactions of the Royal Society of London, 1800. doi: 10.1098/rstl.1800.0018.

HOLMGREN, W; HANSEN, C; MIKOFSKI, M. **pvlib python: a python package for modeling solar energy systems.** Journal of Open Source Software, 3(29), 884, 2018. <https://doi.org/10.21105/joss.00884>

HOTTEL, H. **A Simple Model for Estimating the Transmittance of Direct Solar Radiation through Clear Atmospheres.** Solar Energy, 18(2), 1976. 129-134 p.

HOTTEL, H; WOERTZ, B. **Evaluation of flat-plate solar heat collector.** Trans. ASME, 64, 1942. 91 p.

HUANG, N et al. **The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis.** Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, vol. 454, no. 1971, 1998. 903-995 p.

INPE. **Condições atuais do ENOS: caracterização do El-NIÑO.** 2023. Disponível em: <http://enos.cptec.inpe.br> Acesso em: 30 jun 2023.

IQBAL, M. **An Introduction to Solar Radiation.** Academic Press. 1983.

JACOBSON, M; JADHAV, V. **World estimates of PV optimal tilt angles and ratios of sunlight incident upon tilted and tracked PV panels relative to horizontal panels.** Department of Civil and Environmental Engineering, Stanford University, Stanford, CA 94305-4020, USA, 2018. Disponível em: <https://web.stanford.edu/group/efmh/jacobson/Articles/I/TiltAngles.pdf>. Acesso em: 3 jun 2023.

LEWIS, N. **Toward cost-effective solar energy use.** Science, 315(5813), 2007. 798-801 p.

LI, D, LAM, T; WONG, S. **Lighting and energy performance for an office using high frequency dimming controls.** Energy Conversion and Management, 48(3), 2007. 813-820 p.

LIU, B; JORDAN, R. **The Interrelationship and Characteristic Distribution of Direct, Diffuse and Total Solar Radiation.** Solar Energy, 7(1), 1963. 1-13 p.

LOUTZENHISER, P et al. **Empirical validation of models to compute solar irradiance on inclined surfaces for building energy simulation.** Solar Energy vol. 81., 2007. 254-267 p.

MARTIN, C. **The history of solar energy.** Proceedings of the IEEE, 94(4), 2006. 685-691 p.

MONTEIRO, A; COLLARES-PEREIRA, M. **SUNREL - A Program for Calculating the Solar Radiation Incident on an Inclined Surface.** Renewable Energy, 5(1-4), 1994. 117-120 p.

NREL. **International Data.** 2023. Disponível em: <https://nsrdb.nrel.gov/data-sets/international-data>. Acesso em: 09 jun 2023.

NREL. **Users Manual for TMY3 Data Sets.** 2008. Disponível em: <https://www.nrel.gov/docs/fy08osti/43156.pdf>. Acesso em: 19 jun 2023.

OKAL, E. **On the possibility of seismic recording of meteotsunamis.** Natural Hazards, 104(1), 2020. 81-99 p.

PEREIRA, E; MARTINS, F; GONÇALVES, A; COSTA, R; LIMA, F; RÜTHER, R; ABREU, S; TIEPOLO, G; PEREIRA, S; SOUZA, J. **Atlas brasileiro de energia solar.** 2.ed. São José dos Campos: INPE, 2017. 80 p. Disponível em: <http://doi.org/10.34024/978851700089> Acesso em: 1 jun 2023.

PEREZ, R; CEBECAUER, T; SURI, M. **Semi-Empirical Satellite Models, in: Solar Resource Assessment and Forecasting.** (Editor Jan Kleissl), Elsevier, 2013.

PEREZ, R; INEICHEN, P; KMIECIK, M; MOORE, K; GEORGE, R; RENNÉ, D. **Producing satellite-derived irradiances in complex arid terrain.** Solar Energy 77, 4, 2004. 363-370 p.

PEREZ, R; INEICHEN, P; MOORE, K; KMIECIK, M; CHAIN, C; GEORGE, R; VIGNOLA, F. **A New Operational Satellite-to-Irradiance Model.** Solar Energy 73, 5, 2002. 307-317 p.

PEREZ, R; INEICHEN, P; SEALS, R; MICHALSKY, J; STEWART, R. **Modeling daylight availability and irradiance components from direct and global irradiance.** Solar Energy 44 (5), 1990. 271–289 p.

PEREZ, R; SEALS, R; INEICHEN, P; STEWART, R; MENICUCCI, D. **A new simplified version of the Perez diffuse irradiance model for tilted surfaces.** Solar Energy 39 (3), 1987. 221–232 p.

PEREZ, R; STEWART, R; SEALS, R; GUERTIN, T. **The Development and Verification of the Perez Diffuse Radiation Model.** SAND88-7030. 1988.

PLANCK, M. **Distribution of energy in the normal spectrum.** Verhandlungen der Deutschen Physikalischen Gesellschaft, vol. 2, 1900. 237–245 p.

PULUMBARIT, M. **Choosing a Weather File.** 2023. Disponível em: <https://help-center.helioscope.com/hc/en-us/articles/8316938442259-Choosing-a-Weather-File>. Acesso em: 25 jun 2023.

PVEDUCATION. **Module Structure.** (n.d.). Disponível em: <https://www.pveducation.org/pvcdrom/properties-of-sunlight/properties-of-light>. Acesso em: 3 mai 2023.

PVEDUCATION. **Properties of Light.** (n.d.). Disponível em: <https://www.pveducation.org/pvcdrom/properties-of-sunlight/properties-of-light>. Acesso em: 3 mai 2023.

PVEDUCATION. **Spectral Response.** (n.d.). Disponível em: <https://www.pveducation.org/pvcdrom/solar-cell-operation/spectral-response>. Acesso em: 4 mai 2023.

PVLIB. **Source code for pvlib.irradiance.** (n.d.). Disponível em: https://pvlib-python.readthedocs.io/en/stable/_modules/pvlib/irradiance.html?highlight=pvlib.irradiance. Acesso em: 25 jun 2023.

PVSYST. **NREL's National Solar Radiation Database (NSRD).** (n.d.). Disponível em: https://www.pvsyst.com/help/meteo_source_nrel_nsdb_tmy23.htm. Acesso em: 16 jun 2023.

PVSYST. **PVsyst 7 Help. Transposition Model.** (n.d.). Disponível em: https://www.pvsyst.com/help/models_meteo_transposition.htm. Acesso em: 10 jun 2023.

REDA, I; ANDREAS, A. **Solar position algorithm for solar radiation applications.** Solar Energy, 76(5), 2004. 577–589 p. doi: 10.1016/j.solener.2003.12.003.

RENO, M; HANSEN, C; STEIN, J. **Global Horizontal Irradiance Clear Sky Models: Implementation and Analysis.** Sandia National Laboratories, SAND2012-2389, 2012.

SAKÔ, E; SILVA, J; MESQUITA, D; CAMPOS R; MOREIRA, H; VILLALVA, M. **Concepts and Case Study of Mismatch Losses in Photovoltaic Modules.** In: 15th Brazilian Power Electronics Conference IEEE/COBEP 2019, Santos-SP, 2019.

SANDIA NATIONAL LABORATORIES. **Single Diode Equivalent Circuit Models.** (n.d.). Disponível em: <https://pvpmc.sandia.gov/modeling-steps/2-dc-module-iv/single-diode-equivalent-circuit-models/>. Acesso em: 5 jun 2023.

SENGUPTA, M et al. **Best Practices Handbook for the Collection and Use of Solar Resource Data for Solar Energy Applications.** National Renewable Energy Laboratory (NREL). 2012.

TIAN, J. **A precise series resistance extraction method for crystalline silicon solar cells.** Springer-Verlag Berlin Heidelberg. 2012.

TRENBERTH, K; STEPANIAK, D. **Indices of El Niño Evolution.** Journal of Climate, 14(8), 2001. 1697-1701 p.

UFRGS. **Capítulo 3 – Efeito fotoelétrico.** (n.d.). Disponível em:
https://www.if.ufrgs.br/tex/fis142/fismod/mod03/m_s01.html. Acesso em: 21 jun 2023.

UNIVERSITY OF CALGARY. **Photovoltaic effect.** (n.d.). Disponível em:
https://energyeducation.ca/encyclopedia/Photovoltaic_effect. Acesso em: 18 jun 2023.

VUGRIN, K et al. **Confidence region estimation techniques for nonlinear regression in groundwater flow: Three case studies.** Water Resources Research, Vol. 43, W03423, DOI:10.1029/2005WR004804. (n.d.). Disponível em:
https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html. Acesso em: 20 jun 2023.

WILCOX, S. **National Solar Radiation Database 1991–2010 Update: User’s Manual.** 2012. Disponível em: https://www.nci.noaa.gov/pub/data/nsrdb-solar/documentation-2010/NSRDB_UserManual_r20120906.pdf. Acesso em: 28 jun 2023.

APÊNDICE A - FUNÇÕES ADAPTADAS DO PVLIB (Perez)

```
#Rafael Vilela Santa Rosa
!pip install -q pvlib
import time
import pvlib
from pvlib import location, irradiance, tools
from pvlib.iotools import read_tmy3
import pandas as pd
from matplotlib import pyplot as plt
import pathlib
import numpy as np
from scipy.optimize import curve_fit
from scipy.signal import hilbert
from scipy.stats import skew, kurtosis, variation

tempo_inicial=(time.time())

#obs: arquivo csv escolhido não considera um ano bissexto

energia_hs = [70, 82.4, 122.9, 150.4, 166, 176.9, 174.9, 162.4, 123.8, 102.4, 68.2, 67.5]

df1 = pd.DataFrame({'energia_hs': energia_hs})

energia_pvsyst = [77.9, 88.8, 128.8, 159, 175, 189.1, 185.2, 171.9, 142, 109.1, 85.8, 66]

df2 = pd.DataFrame({'energia_pvsyst': energia_pvsyst})

# get full path to the data directory
DATA_DIR = pathlib.Path(pvlib.__file__).parent / 'data'

# get TMY3 dataset
tmy, metadata = read_tmy3(DATA_DIR / '723170TYA.CSV', coerce_year=2023)

# create location object to store lat, lon, timezone
location = location.Location.from_tmy(metadata)

# print metadata
for key, value in metadata.items():
    print(f'{key}: {value}')
latitude = metadata['latitude']
Φ = latitude

# calculate the necessary variables to do transposition.
times = pd.date_range(start='2023-01-01', end='2023-12-31 23:59:59', freq='1H')

# create a new DataFrame that only contains the necessary columns and that has a consistent time
index
df = pd.DataFrame(index=times)
df['DHI'] = tmy['DHI'].values
df['DNI'] = tmy['DNI'].values
df['GHI'] = tmy['GHI'].values

max_value1 = df['DHI'].max()
print(f"Valor máximo DHI: {max_value1}")
max_value2 = df['DNI'].max()
print(f"Valor máximo DNI: {max_value2}")
max_value3 = df['GHI'].max()
print(f"Valor máximo GHI: {max_value3}")
```

```

solar_position = location.get_solarposition(times)

# calculate extra terrestrial radiation
dni_extra = pvlib.irradiance.get_extra_radiation(df.index)

print('/n')
Optimal_tilt_angle_NH = 1.3793 + Φ*(1.2011 + Φ*(-0.014404 + Φ*0.000080509))
print(f'Inclinação ótima:{Optimal_tilt_angle_NH}')

#####
##### inicio

df_poa = pvlib.irradiance.get_total_irradiance(
    surface_tilt=30, surface_azimuth=90,
    dhi=df['DHI'], dni=df['DNI'], ghi=df['GHI'], dni_extra=dni_extra,
    solar zenith=solar_position['apparent zenith'],
    solar azimuth=solar_position['azimuth'],
    airmass=pvlib.atmosphere.get_relative_airmass(solar_position['apparent zenith']),
    model='Perez')

print("\n-----Dados-----")

df_poa['poa_global'].fillna(0, inplace=True)

# Calculate quartiles
Q1 = df_poa['poa_global'].quantile(0.25)
Q2 = df_poa['poa_global'].quantile(0.5)
Q3 = df_poa['poa_global'].quantile(0.75)
Q4 = df_poa['poa_global'].quantile(1)
P95 = df_poa['poa_global'].quantile(0.95)
P98 = df_poa['poa_global'].quantile(0.98)
P99 = df_poa['poa_global'].quantile(0.99)

print("First Quartile:", Q1)
print("Second Quartile (Median):", Q2)
print("Third Quartile:", Q3)
print("Fourth Quartile (Max):", Q4)
print("95th Percentile:", P95)
print("98th Percentile:", P98)
print("99th Percentile:", P99)
print("\n")

perez_zeros = df_poa['poa_global'][df_poa['poa_global'] == 0].count()
print(f"Number of zeros in Perez model: {perez_zeros}")

perez_high = df_poa['poa_global'][df_poa['poa_global'] > 1367].count()
print(f"Number of high numbers in Perez model: {perez_high}")

perez_mid = df_poa['poa_global'][df_poa['poa_global'] > P98].count()
print(f"Number of mid numbers (>P98) in Perez model: {perez_mid}")

skewness_list = []
envelope_list = []
kurtosis_list = []
cv_list = []
fs=1

# Loop por cada dia do ano
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Get the data for the current day
    day_data = df_poa.loc[day.strftime('%Y-%m-%d'), 'poa_global']

    # Calculate Hilbert transform of the data

```

```

analytic_signal = hilbert(day_data)
amplitude_envelope = np.abs(analytic_signal)
instantaneous_phase = np.unwrap(np.angle(analytic_signal))
instantaneous_frequency = (np.diff(instantaneous_phase) / (2.0*np.pi) * fs)

# Calculate skewness and append to list
skewness_list.append(skew(day_data))

# Calculate mean envelope and append to list
envelope_list.append(np.mean(amplitude_envelope))

# Calculate kurtosis and append to list
kurtosis_list.append(kurtosis(day_data))

# Calculate coefficient of variation and append to list
cv_list.append(variation(day_data))

mean_skewness = np.mean(skewness_list)
mean_envelope = np.mean(envelope_list)
mean_kurtosis = np.mean(kurtosis_list)
mean_cv = np.mean(cv_list)

print(f"Mean skewness: {mean_skewness}")
print(f"Mean envelope: {mean_envelope}")
print(f"Mean kurtosis: {mean_kurtosis}")
print(f"Mean Coefficient of Variation: {mean_cv}")

#print("\nFirst values of Perez model:")
#print(df_poa['poa_global'].head(4000))

# plot the results
df_poa['poa_global'].plot()
plt.ylabel('POA [W/m^2]')
plt.title('POA Irradiance using Perez Model')
plt.show()

# Para um mês específico:
specific_month = df_poa.loc['2023-06']

specific_month['poa_global'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance in 06/2023')
plt.show()

# Para um dia específico:
specific_day = df_poa.loc['2023-06-01']

specific_day['poa_global'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance on 2023-06-01')
plt.show()

##### calculo de energia

# Create a new column for the month
df_poa['month'] = df_poa.index.month

# Group by month and sum the irradiance to get the total irradiance for each month
monthly_irradiance = df_poa.groupby('month')['poa_global'].sum()

```

```

# The total irradiance is in units of W/m^2.
monthly_energy = monthly_irradiance * 3600

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
monthly_energy_kwh = monthly_energy / (1000 * 3600)

percentual1=((monthly_energy_kwh.values/ df1['energia_hs'].values) * 100)
percentual2=((monthly_energy_kwh.values/ df2['energia_pvsyst'].values) * 100)

df3 = pd.DataFrame({'percentual': percentual1})
df4 = pd.DataFrame({'percentual': percentual2})

print('valor médio HS:')
print(df3['percentual'].mean())
print('\nPercentual HelioScope:')
print(df3['percentual'])

print('valor médio PVsyst:')
print(df4['percentual'].mean())
print('\nPercentual PVsyst:')
print(df4['percentual'])

##### identificação de outlier
"""

# Create a new column for the day
df_poa['day'] = df_poa.index.dayofyear

# Group by day and sum the irradiance to get the total irradiance for each day
daily_irradiance = df_poa.groupby('day')['poa_global'].sum()

# The total irradiance is in units of W/m^2.
daily_energy = daily_irradiance * 3600

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
daily_energy_kwh = daily_energy / (1000 * 3600)

"""

# Função para representar a forma senoidal
def sinusoidal(x, a, b, c, d):
    return a * np.sin(b * (x - np.radians(c))) + d

# Agregar os dados por dia
daily_irradiance = df_poa['poa_global'].resample('D').sum()

# Obter o dia do ano para usar como variável independente no ajuste
day_of_year = daily_irradiance.index.dayofyear

# Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
# O período é ajustado para 2*pi/365 para refletir o ciclo anual
# A fase horizontal é ajustada para colocar o pico no meio do ano (dia 182)
initial_parameters = [daily_irradiance.max(), 2 * np.pi / 365, 182, daily_irradiance.min()]

# Ajustar a função aos dados
parameters, _ = curve_fit(sinusoidal, day_of_year, daily_irradiance, p0=initial_parameters)

# Gerar um array de x para a função ajustada
x_fit = np.linspace(1, 365, 1000)

```

```

# Gerar os valores y da função ajustada
y_fit = sinusoidal(x_fit, *parameters)

print(f"A função senoidal de tendência é: {parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - 
{np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}")

# Plotar os dados originais e a função ajustada
plt.scatter(day_of_year, daily_irradiance, label='data')
plt.plot(x_fit, y_fit, color='red', label='fit')
plt.xlabel('Day of year')
plt.ylabel('Irradiação diária [Wh]')
plt.title('Sinusoidal fit to daily POA irradiance data')
plt.legend()
plt.text(170,0,{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - 
{np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}",ha='center', va='center')
plt.show()

##### correção inicial de outlier

print("\n-----Correção inicial outlier-----")

max_value = df_poa['poa_global'].max()
print(f"Valor máximo antes: {max_value}")

df_poa['poa_global'] = df_poa['poa_global'].clip(upper=P98)

max_value = df_poa['poa_global'].max()
print(f"Valor máximo depois: {max_value}")

##### calculo de energia

# Create a new column for the month
df_poa['month'] = df_poa.index.month

# Group by month and sum the irradiance to get the total irradiance for each month
monthly_irradiance = df_poa.groupby('month')['poa_global'].sum()

# The total irradiance is in units of W/m^2.
monthly_energy = monthly_irradiance * 3600

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
monthly_energy_kwh = monthly_energy / (1000 * 3600)

percentual1=((monthly_energy_kwh.values/ df1['energia_hs'].values) * 100)
percentual2=((monthly_energy_kwh.values/ df2['energia_pvsyst'].values) * 100)

df3 = pd.DataFrame({'percentual': percentual1})
df4 = pd.DataFrame({'percentual': percentual2})

print('valor médio HS:')
print(df3['percentual'].mean())
print('\nPercentual HelioScope:')
print(df3['percentual'])

print('valor médio PVsyst:')
print(df4['percentual'].mean())
print('\nPercentual PVsyst:')
print(df4['percentual'])

```

```
#####
##### identificação de outlier

# Função para representar a forma senoidal
def sinusoidal(x, a, b, c, d):
    return a * np.sin(b * (x - np.radians(c))) + d

# Agregar os dados por dia
daily_irradiance = df_poa['poa_global'].resample('D').sum()

# Obter o dia do ano para usar como variável independente no ajuste
day_of_year = daily_irradiance.index.dayofyear

# Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
# O período é ajustado para 2*pi/365 para refletir o ciclo anual
# A fase horizontal é ajustada para colocar o pico no meio do ano (dia 182)
initial_parameters = [daily_irradiance.max(), 2 * np.pi / 365, 182, daily_irradiance.min()]

# Ajustar a função aos dados
parameters, _ = curve_fit(sinusoidal, day_of_year, daily_irradiance, p0=initial_parameters)

# Gerar um array de x para a função ajustada
x_fit = np.linspace(1, 365, 1000)

# Gerar os valores y da função ajustada
y_fit = sinusoidal(x_fit, *parameters)

print(f"A função senoidal de tendência é: {parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}")

# Plotar os dados originais e a função ajustada
plt.scatter(day_of_year, daily_irradiance, label='data')
plt.plot(x_fit, y_fit, color='red', label='fit')
plt.xlabel('Day of year')
plt.ylabel('Irradiação diária [Wh]')
plt.title('Sinusoidal fit to daily POA irradiance data - limitação de potência pelo percentil 98')
plt.legend()
plt.text(170,0,f"{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}", ha='center', va='center')
plt.show()

#####
##### tratamento de outlier

print("\n-----Tratamento de outlier-----")

"""

# Calcule os valores da função de tendência para cada dia do ano
trend_values = sinusoidal(day_of_year, *parameters)

# limite superior e inferior
upper_limit = trend_values * 1.15
lower_limit = trend_values * 1.05

#= série pandas para os valores de tendência e limite superior/inferior para facilitar a manipulação
trend_series = pd.Series(trend_values, index=daily_irradiance.index)
upper_limit_series = pd.Series(upper_limit, index=daily_irradiance.index)
lower_limit_series = pd.Series(lower_limit, index=daily_irradiance.index)

##

# Resample the hourly data to daily data
daily_poa_global = df_poa['poa_global'].resample('D').sum()
```

```

# Replace values in the daily data that are above the upper limit
daily_poa_global.where(daily_poa_global <= upper_limit_series, upper_limit_series, inplace=True)
daily_poa_global.where(daily_poa_global >= lower_limit_series, lower_limit_series, inplace=True)

# Resample the daily data back to hourly data
df_poa['poa_global'] = daily_poa_global.resample('H').ffill() / 24

# plot the results
df_poa['poa_global'].plot()
plt.ylabel('POA [W/m^2]')
plt.title('POA Irradiance using Perez Model')
plt.show()

# Para um mês específico:
specific_month = df_poa.loc['2023-06']

specific_month['poa_global'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance in 06/2023')
plt.show()

# Para um dia específico:
specific_day = df_poa.loc['2023-06-01']

specific_day['poa_global'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance on 2023-06-01')
plt.show()

"""

##### tratamento de outlier 2
# Create a new DataFrame to store the adjusted irradiance values
df_poa_adjusted = df_poa.copy()

daily_parameters = pd.DataFrame(columns=['a', 'b', 'c', 'd'], index=df_poa.index.normalize().unique())

coef_upper = 1.15
coef_lower = 1.05

# Criação de um DataFrame para armazenar os parâmetros diários
daily_parameters = pd.DataFrame(index=pd.date_range('2023-01-01', '2023-12-31'),
                                  columns=['amplitude', 'period', 'phase_shift', 'vertical_shift'])

# Loop por cada dia do ano
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Obter os dados para o dia atual
    day_data = df_poa.loc[day.strftime('%Y-%m-%d')]

    # Obter a hora do dia para usar como variável independente no ajuste
    hour_of_day = day_data.index.hour

    # Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
    # O período é ajustado para 2*pi para refletir o ciclo diário
    initial_parameters = [day_data['poa_global'].max(), 2 * np.pi, 12, day_data['poa_global'].min()]

    # Ajustar a função aos dados
    try:

```

```

parameters, _ = curve_fit(sinusoidal, hour_of_day, day_data['poa_global'],
p0=initial_parameters, maxfev=5000)
except RuntimeError:
    print(f"Failed to fit curve for day {day}")
    continue
# Armazenar os parâmetros no DataFrame
daily_parameters.loc[day, ['amplitude', 'period', 'phase_shift', 'vertical_shift']] = parameters

# Loop por cada dia do ano novamente para ajustar os outliers
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Obter os dados para o dia atual
    day_data = df_poa.loc[day.strftime('%Y-%m-%d')]

    # Obter a hora do dia para usar como variável independente no ajuste
    hour_of_day = day_data.index.hour

    # Obter os parâmetros para o dia atual
    parameters = daily_parameters.loc[day]

    # Calcular os valores da função de tendência para cada hora do dia
    trend_values = sinusoidal(hour_of_day, *parameters)

    # Definir um limite superior e inferior
    upper_limit = trend_values * coef_upper
    lower_limit = trend_values * coef_lower

    # série pandas para os valores de tendência e limite superior/inferior para facilitar a manipulação
    trend_series = pd.Series(trend_values, index=day_data.index)
    upper_limit_series = pd.Series(upper_limit, index=day_data.index)
    lower_limit_series = pd.Series(lower_limit, index=day_data.index)

    for i in range(len(day_data)):
        if day_data['poa_global'].iloc[i] > 0:
            # Substituir valores nos dados diários
            if day_data['poa_global'].iloc[i] > upper_limit_series.iloc[i]:
                day_data['poa_global'].iloc[i] = upper_limit_series.iloc[i]
            elif day_data['poa_global'].iloc[i] < lower_limit_series.iloc[i]:
                day_data['poa_global'].iloc[i] = lower_limit_series.iloc[i]

    # Armazenar os dados ajustados de volta no DataFrame original
    df_poa.loc[day.strftime('%Y-%m-%d'), 'poa_global'] = day_data['poa_global']

# df['total_irradiance'] ajustado para outliers com base na tendência senoidal diária
# para valores negativos por conta do ajuste da senoide
df_poa['poa_global'] = df_poa['poa_global'].clip(lower=0)

print("\n-----Termino do tratamento de outlier-----")

# plot the results
df_poa['poa_global'].plot()
plt.ylabel('POA Irradiance [W/m^2]')
plt.title('POA Irradiance using Perez Model')
plt.show()

# Para um mês específico:
specific_month = df_poa.loc['2023-06']

specific_month['poa_global'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance in 06/2023')
plt.show()

# Para um dia específico:
specific_day = df_poa.loc['2023-06-01']

```

```

specific_day['poa_global'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance on 2023-06-01')
plt.show()

# um dia específico
specific_day = "2023-06-01"

# irradiação para esse dia
day_data = df_poa.loc[specific_day]

# parâmetros para esse dia
parameters = daily_parameters.loc[specific_day]

# curva senoidal para esse dia
hour_of_day = np.arange(24)
sinusoidal_data = sinusoidal(hour_of_day, *parameters)

# irradiação e a curva senoidal
plt.figure(figsize=(10, 6))
plt.plot(day_data.index.hour, day_data['poa_global'], label='Irradiância')
plt.plot(hour_of_day, sinusoidal_data, label='Curva Senoidal', linestyle='--')
plt.xlabel('Hora do dia')
plt.ylabel('Irradiância [W/m²]')
plt.title(f'Irradiância e Curva Senoidal para o dia {specific_day}')
plt.legend()
plt.text(12, 0, f'{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}', ha='center', va='center')
plt.grid(True)
plt.show()

##### identificação de outlier

# Função para representar a forma senoidal
def sinusoidal(x, a, b, c, d):
    return a * np.sin(b * (x - np.radians(c))) + d

# Agregar os dados por dia
daily_irradiance = df_poa['poa_global'].resample('D').sum()

# Obter o dia do ano para usar como variável independente no ajuste
day_of_year = daily_irradiance.index.dayofyear

# Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
# O período é ajustado para 2*pi/365 para refletir o ciclo anual
# A fase horizontal é ajustada para colocar o pico no meio do ano (dia 182)
initial_parameters = [daily_irradiance.max(), 2 * np.pi / 365, 182, daily_irradiance.min()]

# Ajustar a função aos dados
parameters, _ = curve_fit(sinusoidal, day_of_year, daily_irradiance, p0=initial_parameters)

# Gerar um array de x para a função ajustada
x_fit = np.linspace(1, 365, 1000)

# Gerar os valores y da função ajustada
y_fit = sinusoidal(x_fit, *parameters)

print(f"A função senoidal de tendência é: {parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}")

# Plotar os dados originais e a função ajustada

```

```

plt.scatter(day_of_year, daily_irradiance, label='data')
plt.plot(x_fit, y_fit, color='red', label='fit')
plt.xlabel('Day of year')
plt.ylabel('Irradiância diária [Wh]')
plt.title('Sinusoidal fit to daily POA irradiance data - com tratamento de outliers')
plt.legend()
plt.text(170,0,f'{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x -
{np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}",ha='center', va='center')
plt.show()

#####
##### calculo de energia

# Create a new column for the month
df_poa['month'] = df_poa.index.month

# Group by month and sum the irradiance to get the total irradiance for each month
monthly_irradiance = df_poa.groupby('month')['poa_global'].sum()

# The total irradiance is in units of W/m^2.
monthly_energy = monthly_irradiance * 3600

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
monthly_energy_kwh = monthly_energy / (1000 * 3600)

#monthly_energy_kwh

percentual3=((monthly_energy_kwh.values/ df1['energia_hs'].values) * 100)
percentual4=((monthly_energy_kwh.values/ df2['energia_pvsyst'].values) * 100)

df5 = pd.DataFrame({'percentual': percentual3})
df6 = pd.DataFrame({'percentual': percentual4})

print('valor médio HS:')
print(df5['percentual'].mean())
print('\nPercentual HelioScope:')
print(df5['percentual'])
print('valor médio PVsyst:')
print(df6['percentual'].mean())
print('\nPercentual PVsyst:')
print(df6['percentual'])

months = list(range(1, 13))
errors1 = np.abs(df3['percentual'] - 100)
errors2 = np.abs(df4['percentual'] - 100)
errors3 = np.abs(df5['percentual'] - 100)
errors4 = np.abs(df6['percentual'] - 100)

# Plot HelioScope data
plt.errorbar(months, df3['percentual'], yerr=errors1, fmt='o', color='blue', label='Original (em relação
ao HelioScope)')

plt.errorbar(months, df5['percentual'], yerr=errors3, fmt='o', color='red', label='Com correção de
outlier (em relação ao HelioScope)')

# linha em y=100
plt.axhline(100, color='black', linestyle='--')

plt.title('Comparação de percentuais mensais (Perez/HelioScope)')
plt.xlabel('Mês')
plt.ylabel('Percentual (%)')

```

```

plt.legend()

for month, error1, error3 in zip(months, errors1, errors3):
    plt.text(month, df3['percentual'][month-1], f'{error1:.1f}%', ha='center', va='bottom')
    plt.text(month, df5['percentual'][month-1], f'{error3:.1f}%', ha='center', va='top')

plt.legend()

# Exibir o gráfico
plt.show()

#####
# Plot PVsyst data
plt.errorbar(months, df4['percentual'], yerr=errors2, fmt='o', color='yellow', label='Original (em
relação ao PVsyst)')
plt.errorbar(months, df6['percentual'], yerr=errors4, fmt='o', color='green', label='Com correção de
outlier (em relação ao PVsyst)')

# linha em y=100
plt.axhline(100, color='black', linestyle='--')

plt.title('Comparação de percentuais mensais (Perez/PVsyst)')
plt.xlabel('Mês')
plt.ylabel('Percentual (%)')

plt.legend()

for month, error2, error4 in zip(months, errors2, errors4):
    plt.text(month, df4['percentual'][month-1], f'{error2:.1f}%', ha='center', va='bottom')
    plt.text(month, df6['percentual'][month-1], f'{error4:.1f}%', ha='center', va='top')

plt.legend()

# Exibir o gráfico
plt.show()

skewness_list2 = []
envelope_list2 = []
kurtosis_list2 = []
cv_list2 = []

# Loop por cada dia do ano
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Get the data for the current day
    day_data = df_poa.loc[day.strftime('%Y-%m-%d'), 'poa_global']

    # Calculate Hilbert transform of the data
    analytic_signal = hilbert(day_data)
    amplitude_envelope = np.abs(analytic_signal)
    instantaneous_phase = np.unwrap(np.angle(analytic_signal))
    instantaneous_frequency = (np.diff.instantaneous_phase) / (2.0*np.pi) * fs

    # Calculate skewness and append to list
    skewness_list2.append(skew(day_data))

    # Calculate mean envelope and append to list
    envelope_list2.append(np.mean(amplitude_envelope))

    # Calculate kurtosis and append to list
    kurtosis_list2.append(kurtosis(day_data))

```

```
# Calculate coefficient of variation and append to list
cv_list2.append(variation(day_data))

mean_skewness = np.mean(skewness_list2)
mean_envelope = np.mean(envelope_list2)
mean_kurtosis = np.mean(kurtosis_list2)
mean_cv = np.mean(cv_list2)

print(f"Mean skewness: {mean_skewness}")
print(f"Mean envelope: {mean_envelope}")
print(f"Mean kurtosis: {mean_kurtosis}")
print(f"Mean Coefficient of Variation: {mean_cv}")

tempo_final=(time.time())
tempo=tempo_final - tempo_inicial

print(f"{tempo} segundos")
print("Fim!")
```

APÊNDICE B - FUNÇÕES ADAPTADAS DO PVLIB (Hay-Davies)

```
#Rafael Vilela Santa Rosa
!pip install -q pvlib
import time
import pvlib
from pvlib import location, irradiance, tools
from pvlib.iotools import read_tmy3
import pandas as pd
from matplotlib import pyplot as plt
import pathlib
import numpy as np
from scipy.optimize import curve_fit
from scipy.signal import hilbert
from scipy.stats import skew, kurtosis, variation

tempo_inicial=(time.time())

#obs: arquivo csv escolhido não considera um ano bissexto
energia_hs = [70, 82.4, 122.9, 150.4, 166, 176.9, 174.9, 162.4, 123.8, 102.4, 68.2, 67.5]
df1 = pd.DataFrame({'energia_hs': energia_hs})

# get full path to the data directory
DATA_DIR = pathlib.Path(pvlib.__file__).parent / 'data'

# get TMY3 dataset
#tmy, metadata = read_tmy3(DATA_DIR / '703165TY.csv', coerce_year=2023)
tmy, metadata = read_tmy3(DATA_DIR / '723170TYA.CSV', coerce_year=2023)

# create location object to store lat, lon, timezone
location = location.Location.from_tmy(metadata)

# print metadata
for key, value in metadata.items():
    print(f'{key}: {value}')
latitude = metadata['latitude']
Φ = latitude

# calculate the necessary variables to do transposition.
times = pd.date_range(start='2023-01-01', end='2023-12-31 23:59:59', freq='1H')

# create a new DataFrame that only contains the necessary columns and that has a consistent time
index
df = pd.DataFrame(index=times)
df['DHI'] = tmy['DHI'].values
df['DNI'] = tmy['DNI'].values
df['GHI'] = tmy['GHI'].values

max_value1 = df['DHI'].max()
print(f"Valor máximo DHI: {max_value1}")
max_value2 = df['DNI'].max()
print(f"Valor máximo DNI: {max_value2}")
max_value3 = df['GHI'].max()
print(f"Valor máximo GHI: {max_value3}")

solar_position = location.get_solarposition(times)
```

```
# calculate extra terrestrial radiation  
dni_extra = pvlib.irradiance.get_extra_radiation(df.index)
```

```
Optimal_tilt_angle_NH = 1.3793 + Φ*(1.2011 + Φ*(-0.014404 + Φ*0.000080509))  
print(f'Inclinação ótima:{Optimal_tilt_angle_NH}')
```

```
##### inicio
```

```
##### beam
```

```
def aoi_projection(surface_tilt, surface_azimuth, solar zenith, solar azimuth):
```

```
projection = (
    tools.cosd(surface_tilt) * tools.cosd(solar_zenith) +
    tools.sind(surface_tilt) * tools.sind(solar_zenith) *
    tools.cosd(solar_azimuth - surface_azimuth))
```

```
# GH 1185  
projection = np.clip(projection, -1, 1)
```

```
try:  
    projection.name = 'aoi_projection'  
except AttributeError:  
    pass
```

return projection

```
def beam_component(surface_tilt, surface_azimuth, solar zenith, solar azimuth,  
dni):
```

```
beam = dni * aoi_projection(surface_tilt, surface_azimuth,  
                           solar zenith, solar azimuth)  
beam = np.maximum(beam, 0)
```

return beam

```
df['beam']=beam_component(surface_tilt=30, surface_azimuth=90,  
solar_zenith=solar_position['apparent_zenith'], solar_azimuth=solar_position['azimuth'],  
dni=df['DNI'])
```

group

```
SURFACE_ALBEDOS = {'urban': 0.18,  
                     'grass': 0.20,  
                     'fresh grass': 0.26,  
                     'soil': 0.17,  
                     'sand': 0.40,  
                     'snow': 0.65,  
                     'fresh snow': 0.75,  
                     'asphalt': 0.12,  
                     'concrete': 0.30,  
                     'aluminum': 0.85,  
                     'copper': 0.74,  
                     'fresh steel': 0.35,  
                     'dirty steel': 0.08,  
                     'sea': 0.06}
```

```
df['ground_reflected']= pvlib.irradiance.get_ground_diffuse(surface_tilt=30, ghi=df['GHI'], albedo=0.2, surface_type=None)
```

```
##### diffuse

# Hay Davies
df['haydavies_diffuse'] = pvlib.irradiance.haydavies(surface_tilt=30, surface_azimuth=270,
                                                       dhi=df['DHI'], dni=df['DNI'], dni_extra=dni_extra,
                                                       solar zenith=solar_position['apparent zenith'],
                                                       solar azimuth=solar_position['azimuth'])

print("\n-----Dados-----")

beam_zeros = df['beam'][df['beam'] == 0].count()
print(f"Number of zeros in beam model: {beam_zeros}")

beam_high = df['beam'][df['beam'] > 1367].count()
print(f"Number of high numbers (>1367) in beam model: {beam_high}")

# plot the results
df['beam'].plot()
plt.ylabel('Beam Irradiance [W/m^2]')
plt.title('Beam Model')
plt.show()

ground_zeros = df['ground_reflected'][df['ground_reflected'] == 0].count()
print(f"Number of zeros in GR: {ground_zeros}")

# plot the results
df['ground_reflected'].plot()
plt.ylabel('Ground Reflected Irradiance [W/m^2]')
plt.title('Ground Reflected Model')
plt.show()

haydavies_diffuse_zeros = df['haydavies_diffuse'][df['haydavies_diffuse'] == 0].count()
print(f"Number of zeros in haydavies_diffuse model: {haydavies_diffuse_zeros}")

# plot the results
df['haydavies_diffuse'].plot()
plt.ylabel('Diffuse Sky Irradiance [W/m^2]')
plt.title('Diffuse Sky Irradiance using Hay Davies Model')
plt.show()

##### somatorio

print("\n-----Dados agrupados-----")

df['total_irradiance'] = df['haydavies_diffuse'] + df['ground_reflected'] + df['beam']

skewness_list = []
envelope_list = []
kurtosis_list = []
cv_list = []
fs=1

# Loop por cada dia do ano
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Get the data for the current day
    day_data = df.loc[day.strftime('%Y-%m-%d'), 'total_irradiance']

    # Calculate Hilbert transform of the data
    analytic_signal = hilbert(day_data)
```

```

amplitude_envelope = np.abs(analytic_signal)
instantaneous_phase = np.unwrap(np.angle(analytic_signal))
instantaneous_frequency = (np.diff(instantaneous_phase) / (2.0*np.pi) * fs)

# Calculate skewness and append to list
skewness_list.append(skew(day_data))

# Calculate mean envelope and append to list
envelope_list.append(np.mean(amplitude_envelope))

# Calculate kurtosis and append to list
kurtosis_list.append(kurtosis(day_data))

# Calculate coefficient of variation and append to list
cv_list.append(variation(day_data))

mean_skewness = np.mean(skewness_list)
mean_envelope = np.mean(envelope_list)
mean_kurtosis = np.mean(kurtosis_list)
mean_cv = np.mean(cv_list)

print(f"Mean skewness: {mean_skewness}")
print(f"Mean envelope: {mean_envelope}")
print(f"Mean kurtosis: {mean_kurtosis}")
print(f"Mean Coefficient of Variation: {mean_cv}")

# Calculate quartiles
Q1 = df['total_irradiance'].quantile(0.25)
Q2 = df['total_irradiance'].quantile(0.5)
Q3 = df['total_irradiance'].quantile(0.75)
Q4 = df['total_irradiance'].quantile(1)
P95 = df['total_irradiance'].quantile(0.95)
P98 = df['total_irradiance'].quantile(0.98)
P99 = df['total_irradiance'].quantile(0.99)

print("First Quartile:", Q1)
print("Second Quartile (Median):", Q2)
print("Third Quartile:", Q3)
print("Fourth Quartile (Max):", Q4)
print("95th Percentile:", P95)
print("98th Percentile:", P98)
print("99th Percentile:", P99)
print("\n")

total_irradiance_zeros = df['total_irradiance'][df['total_irradiance'] == 0].count()
print(f"Number of zeros in total_irradiance: {total_irradiance_zeros}")

haydavies_high = df['total_irradiance'][df['total_irradiance'] > 1367].count()
print(f"Number of high numbers in haydavies model: {haydavies_high}")

haydavies_mid = df['total_irradiance'][df['total_irradiance'] > P98].count()
print(f"Number of mid numbers (>P99) in haydavies model: {haydavies_mid}")

##### valores sem tratamento de outlier

# Create a new column for the month
df['month'] = df.index.month

# Group by month and sum the irradiance to get the total irradiance for each month
monthly_irradiance = df.groupby('month')['total_irradiance'].sum()

# The total irradiance is in units of W/m^2.
monthly_energy = monthly_irradiance * 3600

```

```

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
monthly_energy_kwh = monthly_energy / (1000 * 3600)

# Calcula a diferença percentual
percentual1=((monthly_energy_kwh.values/ df1['energia_hs'].values) * 100)
df2 = pd.DataFrame({'percentual': percentual1})

print('valor médio HS:')
print(df2['percentual'].mean())
print('\nPercentual HelioScope:')
print(df2['percentual'])

##### continuaçāo

# plot the results
df['total_irradiance'].plot()
plt.ylabel('POA Irradiance [W/m^2]')
plt.title('POA Irradiance using Hay Davies Model')
plt.show()

# Para um mēs específico:
specific_month = df.loc['2023-06']

specific_month['total_irradiance'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance in 06/2023')
plt.show()

# Para um dia específico:
specific_day = df.loc['2023-06-01']

specific_day['total_irradiance'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance on 2023-06-01')
plt.show()

##### correção inicial de outlier, se necessário

#df['total_irradiance'] = df['total_irradiance'].clip(upper=P98)

max_value = df['total_irradiance'].max()
print(f"Valor māximo: {max_value}")

##### identificaçāo de outlier

"""

# Create a new column for the day
df['day'] = df.index.dayofyear

# Group by day and sum the irradiance to get the total irradiance for each day
daily_irradiance = df.groupby('day')['total_irradiance'].sum()

# The total irradiance is in units of W/m^2. To convert this to energy, we multiply by the number of
# seconds in an hour (3600),
# since each row in the DataFrame represents an hour.

```

```

daily_energy = daily_irradiance * 3600

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
daily_energy_kwh = daily_energy / (1000 * 3600)

"""

# Função para representar a forma senoidal
def sinusoidal(x, a, b, c, d):
    return a * np.sin(b * (x - np.radians(c))) + d

# Agregar os dados por dia
daily_irradiance = df['total_irradiance'].resample('D').sum()

# Obter o dia do ano para usar como variável independente no ajuste
day_of_year = daily_irradiance.index.dayofyear

# Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
# O período é ajustado para 2*pi/365 para refletir o ciclo anual
# A fase horizontal é ajustada para colocar o pico no meio do ano (dia 182)
initial_parameters = [daily_irradiance.max(), 2 * np.pi / 365, 182, daily_irradiance.min()]

# Ajustar a função aos dados
parameters, _ = curve_fit(sinusoidal, day_of_year, daily_irradiance, p0=initial_parameters)

# Gerar um array de x para a função ajustada
x_fit = np.linspace(1, 365, 1000)

# Gerar os valores y da função ajustada
y_fit = sinusoidal(x_fit, *parameters)

print(f"A função senoidal de tendência é: {parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}")

# Plotar os dados originais e a função ajustada
plt.scatter(day_of_year, daily_irradiance, label='data')
plt.plot(x_fit, y_fit, color='red', label='fit')
plt.xlabel('Day of year')
plt.ylabel('Irradiation diária [Wh]')
plt.title('Sinusoidal fit to daily POA irradiance data')
plt.legend()
plt.text(170, 500, f'{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}", ha='center', va='center')
plt.show()

#####
##### tratamento outlier

print("\n-----Tratamento de outlier-----")

"""

# valores da função de tendência para cada dia do ano
trend_values = sinusoidal(day_of_year, *parameters)

# limite superior e inferior
upper_limit = trend_values * 1.20
lower_limit = trend_values * 1.05

# série pandas para os valores de tendência e limite superior/inferior para facilitar a manipulação
trend_series = pd.Series(trend_values, index=daily_irradiance.index)
upper_limit_series = pd.Series(upper_limit, index=daily_irradiance.index)
lower_limit_series = pd.Series(lower_limit, index=daily_irradiance.index)

```

```

## 

# Resample the hourly data to daily data
daily_poa_global = df['total_irradiance'].resample('D').sum()

# Replace values in the daily data that are above the upper limit
daily_poa_global.where(daily_poa_global <= upper_limit_series, upper_limit_series, inplace=True)
daily_poa_global.where(daily_poa_global >= lower_limit_series, lower_limit_series, inplace=True)

# Resample the daily data back to hourly data
df['total_irradiance'] = daily_poa_global.resample('H').ffill() / 24

# plot the results
df['total_irradiance'].plot()
plt.ylabel('POA [W/m^2]')
plt.title('POA Irradiance using haydavies Model')
plt.show()

# Para um mês específico:
specific_month = df.loc['2023-06']

specific_month['total_irradiance'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance in 06/2023')
plt.show()

# Para um dia específico:
specific_day = df.loc['2023-06-01']

specific_day['total_irradiance'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance on 2023-06-01')
plt.show()

"""

##### tratamento outlier 2

# Create a new DataFrame to store the adjusted irradiance values
df_adjusted = df.copy()

daily_parameters = pd.DataFrame(columns=['a', 'b', 'c', 'd'], index=df.index.normalize().unique())

coef_upper = 1.20
coef_lower = 1.05

# Criação de um DataFrame para armazenar os parâmetros diários
daily_parameters = pd.DataFrame(index=pd.date_range('2023-01-01', '2023-12-31'),
                                 columns=['amplitude', 'period', 'phase_shift', 'vertical_shift'])

# Loop por cada dia do ano
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Obter os dados para o dia atual
    day_data = df.loc[day.strftime('%Y-%m-%d')]

    # Obter a hora do dia para usar como variável independente no ajuste
    hour_of_day = day_data.index.hour

    # Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
    # O período é ajustado para 2*pi para refletir o ciclo diário

```

```

initial_parameters = [day_data['total_irradiance'].max(), 2 * np.pi, 12,
day_data['total_irradiance'].min()]

# Ajustar a função aos dados
try:
    parameters, _ = curve_fit(sinusoidal, hour_of_day, day_data['total_irradiance'],
p0=initial_parameters, maxfev=5000)
except RuntimeError:
    print(f"Failed to fit curve for day {day}")
    continue
# Armazenar os parâmetros no DataFrame
daily_parameters.loc[day, ['amplitude', 'period', 'phase_shift', 'vertical_shift']] = parameters

# Loop por cada dia do ano novamente para ajustar os outliers
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Obter os dados para o dia atual
    day_data = df.loc[day.strftime('%Y-%m-%d')]

    # Obter a hora do dia para usar como variável independente no ajuste
    hour_of_day = day_data.index.hour

    # Obter os parâmetros para o dia atual
    parameters = daily_parameters.loc[day]

    # Calcular os valores da função de tendência para cada hora do dia
    trend_values = sinusoidal(hour_of_day, *parameters)

    # Definir um limite superior e inferior
    upper_limit = trend_values * coef_upper
    lower_limit = trend_values * coef_lower

    # série pandas para os valores de tendência e limite superior/inferior para facilitar a manipulação
    trend_series = pd.Series(trend_values, index=day_data.index)
    upper_limit_series = pd.Series(upper_limit, index=day_data.index)
    lower_limit_series = pd.Series(lower_limit, index=day_data.index)

    for i in range(len(day_data)):
        if day_data['total_irradiance'].iloc[i] > 0:
            # Substituir valores nos dados diários
            if day_data['total_irradiance'].iloc[i] > upper_limit_series.iloc[i]:
                day_data['total_irradiance'].iloc[i] = upper_limit_series.iloc[i]
            elif day_data['total_irradiance'].iloc[i] < lower_limit_series.iloc[i]:
                day_data['total_irradiance'].iloc[i] = lower_limit_series.iloc[i]

    # Armazenar os dados ajustados de volta no DataFrame original
    df.loc[day.strftime('%Y-%m-%d'), 'total_irradiance'] = day_data['total_irradiance']

# df['total_irradiance'] ajustado para outliers com base na tendência senoidal diária

# para valores negativos por conta do ajuste da senoide
df['total_irradiance'] = df['total_irradiance'].clip(lower=0)

print("\n-----Termino de tratamento de outlier-----")

# plot the results
df['total_irradiance'].plot()
plt.ylabel('POA Irradiance [W/m^2]')
plt.title('POA Irradiance using Hay Davies Model')
plt.show()

# Para um mês específico:
specific_month = df.loc['2023-06']

```

```

specific_month['total_irradiance'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance in 06/2023')
plt.show()

# Para um dia específico:
specific_day = df.loc['2023-06-01']

specific_day['total_irradiance'].plot()
plt.ylabel('Irradiance [W/m^2]')
plt.title('POA Irradiance on 2023-06-01')
plt.show()

# um dia específico
specific_day = "2023-08-02"

# Obtenha a irradiação para esse dia
day_data = df.loc[specific_day]

# Obtenha os parâmetros para esse dia
parameters = daily_parameters.loc[specific_day]

# Gere a curva senoidal para esse dia
hour_of_day = np.arange(24)
sinusoidal_data = sinusoidal(hour_of_day, *parameters)

# Plote a irradiação e a curva senoidal
plt.figure(figsize=(10, 6))
plt.plot(day_data.index.hour, day_data['total_irradiance'], label='Irradiância')
plt.plot(hour_of_day, sinusoidal_data, label='Curva Senoidal', linestyle='--')
plt.xlabel('Hora do dia')
plt.ylabel('Irradiância [W/m^2]')
plt.title(f'Irradiância e Curva Senoidal para o dia {specific_day}')
plt.legend()
plt.text(12, 0, f'{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x - {np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}', ha='center', va='center')
plt.grid(True)
plt.show()

##### identificação outlier

# Função para representar a forma senoidal
def sinusoidal(x, a, b, c, d):
    return a * np.sin(b * (x - np.radians(c))) + d

# Agregar os dados por dia
daily_irradiance = df['total_irradiance'].resample('D').sum()

# Obter o dia do ano para usar como variável independente no ajuste
day_of_year = daily_irradiance.index.dayofyear

# Parâmetros iniciais para o ajuste: amplitude, período, fase horizontal, fase vertical
# O período é ajustado para 2*pi/365 para refletir o ciclo anual
# A fase horizontal é ajustada para colocar o pico no meio do ano (dia 182)
initial_parameters = [daily_irradiance.max(), 2 * np.pi / 365, 182, daily_irradiance.min()]

# Ajustar a função aos dados
parameters, _ = curve_fit(sinusoidal, day_of_year, daily_irradiance, p0=initial_parameters)

# Gerar um array de x para a função ajustada
x_fit = np.linspace(1, 365, 1000)

# Gerar os valores y da função ajustada
y_fit = sinusoidal(x_fit, *parameters)

```

```

print(f"A função senoidal de tendência é: {parameters[0]:.2f} * sin({parameters[1]:.2f} * (x -
{np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}")

# Plotar os dados originais e a função ajustada
plt.scatter(day_of_year, daily_irradiance, label='data')
plt.plot(x_fit, y_fit, color='red', label='fit')
plt.xlabel('Day of year')
plt.ylabel('Irradiação diária [Wh]')
plt.title('Sinusoidal fit to daily POA irradiance data - com tratamento de outliers')
plt.legend()
plt.text(170,500,f"{parameters[0]:.2f} * sin({parameters[1]:.2f} * (x -
{np.degrees(parameters[2]):.2f})) + {parameters[3]:.2f}",ha='center', va='center')
plt.show()

#####
##### calculo de energia

# Create a new column for the month
df['month'] = df.index.month

# Group by month and sum the irradiance to get the total irradiance for each month
monthly_irradiance = df.groupby('month')['total_irradiance'].sum()

# The total irradiance is in units of W/m^2.
monthly_energy = monthly_irradiance * 3600

# Convert the energy from Joules (J) to kilowatt-hours (kWh)
monthly_energy_kwh = monthly_energy / (1000 * 3600)

#monthly_energy_kwh

# Calcula a diferença percentual

percentual2=((monthly_energy_kwh.values/ df1['energia_hs'].values) * 100)

df3 = pd.DataFrame({'percentual': percentual2})

print('valor médio HS:')
print(df3['percentual'].mean())
print('\nPercentual HelioScope:')
print(df3['percentual'])

months = list(range(1, 13))
errors1 = np.abs(df2['percentual'] - 100)
errors2 = np.abs(df3['percentual'] - 100)

# Plot HelioScope data
plt.errorbar(months, df2['percentual'], yerr=errors1, fmt='o', color='blue', label='Original (em relação
ao HelioScope)')

# Plot PVsyst data
plt.errorbar(months, df3['percentual'], yerr=errors2, fmt='o', color='red', label='Com correção de
outlier (em relação ao HelioScope)')

# Adicione a linha em y=100
plt.axhline(100, color='black', linestyle='--')

# Adicione títulos e rótulos
plt.title('Comparação de percentuais mensais (Hay-Davies/HelioScope)')
plt.xlabel('Mês')
plt.ylabel('Percentual (%)')

```

```

# Adicione uma legenda
plt.legend()

for month, error1, error2 in zip(months, errors1, errors2):
    plt.text(month, df2['percentual'][month-1], f'{error1:.1f}%', ha='center', va='bottom')
    plt.text(month, df3['percentual'][month-1], f'{error2:.1f}%', ha='center', va='top')

# Exibir o gráfico
plt.show()

skewness_list2 = []
envelope_list2 = []
kurtosis_list2 = []
cv_list2 = []

# Loop por cada dia do ano
for day in pd.date_range('2023-01-01', '2023-12-31'):
    # Get the data for the current day
    day_data = df.loc[day.strftime('%Y-%m-%d'), 'total_irradiance']

    # Calculate Hilbert transform of the data
    analytic_signal = hilbert(day_data)
    amplitude_envelope = np.abs(analytic_signal)
    instantaneous_phase = np.unwrap(np.angle(analytic_signal))
    instantaneous_frequency = (np.diff.instantaneous_phase) / (2.0*np.pi) * fs

    # Calculate skewness and append to list
    skewness_list2.append(skew(day_data))

    # Calculate mean envelope and append to list
    envelope_list2.append(np.mean(amplitude_envelope))

    # Calculate kurtosis and append to list
    kurtosis_list2.append(kurtosis(day_data))

    # Calculate coefficient of variation and append to list
    cv_list2.append(variation(day_data))

mean_skewness = np.mean(skewness_list2)
mean_envelope = np.mean(envelope_list2)
mean_kurtosis = np.mean(kurtosis_list2)
mean_cv = np.mean(cv_list2)

print(f"Mean skewness: {mean_skewness}")
print(f"Mean envelope: {mean_envelope}")
print(f"Mean kurtosis: {mean_kurtosis}")
print(f"Mean Coefficient of Variation: {mean_cv}")

tempo_final=(time.time())
tempo= tempo_final - tempo_inicial

print(f"{tempo} segundos")
print("Fim!")

```

ANEXO A – BIBLIOTECA PVLIB E SUAS PRINCIPAIS FUNÇÕES UTILIZADAS

"""

The ``irradiance`` module contains functions for modeling global horizontal irradiance, direct normal irradiance, diffuse horizontal irradiance, and total irradiance under various conditions.

"""

```
import datetime
from collections import OrderedDict
from functools import partial

import numpy as np
import pandas as pd

from pvlib import atmosphere, solarposition, tools
import pvlib # used to avoid dni name collision in complete_irradiance
```

#####

```
def get_total_irradiance(surface_tilt, surface_azimuth,
                        solar zenith, solar azimuth,
                        dni, ghi, dhi, dni_extra=None, airmass=None,
                        albedo=0.25, surface_type=None,
                        model='isotropic',
                        model_perez='allsitescomposite1990'):
```

r"""

Determine total in-plane irradiance and its beam, sky diffuse and ground reflected components, using the specified sky diffuse irradiance model.

.. math:::

$$I_{\text{tot}} = I_{\text{beam}} + I_{\text{sky diffuse}} + I_{\text{ground}}$$

Sky diffuse models include:

- * isotropic (default)
- * klucher
- * haydavies
- * reindl
- * king
- * perez

Parameters

```
surface_tilt : numeric
    Panel tilt from horizontal. [degree]
surface_azimuth : numeric
    Panel azimuth from north. [degree]
solar zenith : numeric
    Solar zenith angle. [degree]
solar azimuth : numeric
    Solar azimuth angle. [degree]
dni : numeric
    Direct Normal Irradiance. [W/m2]
ghi : numeric
    Global horizontal irradiance. [W/m2]
dhi : numeric
    Diffuse horizontal irradiance. [W/m2]
dni_extra : None or numeric, default None
    Extraterrestrial direct normal irradiance. [W/m2]
```

airmass : None or numeric, default None
 Relative airmass (not adjusted for pressure). [unitless]
 albedo : numeric, default 0.25
 Ground surface albedo. [unitless]
 surface_type : None or str, default None
 Surface type. See :py:func:`~pvlib.irradiance.get_ground_diffuse` for
 the list of accepted values.
 model : str, default 'isotropic'
 Irradiance model. Can be one of ``'isotropic'``, ``'klucher'``,
 ``'haydavies'``, ``'reindl'``, ``'king'``, ``'perez'``.
 model_perez : str, default 'allsitescomposite1990'
 Used only if ``model='perez'``. See :py:func:`~pvlib.irradiance.perez`.

Returns

 total_irrad : OrderedDict or DataFrame
 Contains keys/columns ``'poa_global', 'poa_direct', 'poa_diffuse',
 'poa_sky_diffuse', 'poa_ground_diffuse'``.

Notes

 Models ``'haydavies'``, ``'reindl'``, or ``'perez'`` require
 ``'dni_extra'``. Values can be calculated using
 :py:func:`~pvlib.irradiance.get_extra_radiation`.

The ``'perez'`` model requires relative airmass (``airmass``) as input. If
 ``airmass`` is not provided, it is calculated using the defaults in
 :py:func:`~pvlib.atmosphere.get_relative_airmass`.
 """

```
poa_sky_diffuse = get_sky_diffuse(  

    surface_tilt, surface_azimuth, solar zenith, solar_azimuth,  

    dni, ghi, dhi, dni_extra=dni_extra, airmass=airmass, model=model,  

    model_perez=model_perez)  
  

poa_ground_diffuse = get_ground_diffuse(surface_tilt, ghi, albedo,  

                                         surface_type)  

aoi_ = aoi(surface_tilt, surface_azimuth, solar zenith, solar_azimuth)  

irrads = poa_components(aoi_, dni, poa_sky_diffuse, poa_ground_diffuse)  

return irrads
```

#####

```
def get_extra_radiation(datetime_or_doy, solar_constant=1366.1,  

                       method='spencer', epoch_year=2014, **kwargs):  

    """  

    Determine extraterrestrial radiation from day of year.
```

Parameters

 datetime_or_doy : numeric, array, date, datetime, Timestamp, DatetimeIndex
 Day of year, array of days of year, or datetime-like object

solar_constant : float, default 1366.1
 The solar constant.

method : string, default 'spencer'
 The method by which the ET radiation should be calculated.
 Options include ``'pyephem', 'spencer', 'asce', 'nrel'``.

epoch_year : int, default 2014
 The year in which a day of year input will be calculated. Only
 applies to day of year input used with the pyephem or nrel

methods.

kwargs :
Passed to solarposition.nrel_earthsun_distance

Returns

dni_extra : float, array, or Series

The extraterrestrial radiation present in watts per square meter on a surface which is normal to the sun. Pandas Timestamp and DatetimeIndex inputs will yield a Pandas TimeSeries. All other inputs will yield a float or an array of floats.

References

.. [1] M. Reno, C. Hansen, and J. Stein, "Global Horizontal Irradiance Clear Sky Models: Implementation and Analysis", Sandia National Laboratories, SAND2012-2389, 2012.

.. [2] <<http://solardat.uoregon.edu/SolarRadiationBasics.html>>, Eqs. SR1 and SR2

.. [3] Partridge, G. W. and Platt, C. M. R. 1976. Radiative Processes in Meteorology and Climatology.

.. [4] Duffie, J. A. and Beckman, W. A. 1991. Solar Engineering of Thermal Processes, 2nd edn. J. Wiley and Sons, New York.

.. [5] ASCE, 2005. The ASCE Standardized Reference Evapotranspiration Equation, Environmental and Water Resources Institute of the American Civil Engineers, Ed. R. G. Allen et al.

"""

```
to_doy, to_datetimeindex, to_output = \
    _handle_extra_radiation_types(datetime_or_doy, epoch_year)

# consider putting asce and spencer methods in their own functions
method = method.lower()
if method == 'asce':
    B = solarposition._calculate_simple_day_angle(to_doy(datetime_or_doy),
                                                   offset=0)
    RoverR0sqrd = 1 + 0.033 * np.cos(B)
elif method == 'spencer':
    B = solarposition._calculate_simple_day_angle(to_doy(datetime_or_doy))
    RoverR0sqrd = (1.00011 + 0.034221 * np.cos(B) + 0.00128 * np.sin(B) +
                   0.000719 * np.cos(2 * B) + 7.7e-05 * np.sin(2 * B))
elif method == 'pyephem':
    times = to_datetimeindex(datetime_or_doy)
    RoverR0sqrd = solarposition.pyephem_earthsun_distance(times) ** (-2)
elif method == 'nrel':
    times = to_datetimeindex(datetime_or_doy)
    RoverR0sqrd = \
        solarposition.nrel_earthsun_distance(times, **kwargs) ** (-2)
else:
    raise ValueError('Invalid method: %s', method)

Ea = solar_constant * RoverR0sqrd

Ea = to_output(Ea)

return Ea

#####
```

```
def get_ground_diffuse(surface_tilt, ghi, albedo=.25, surface_type=None):
```

```
    """  
    Estimate diffuse irradiance from ground reflections given  
    irradiance, albedo, and surface tilt.
```

Function to determine the portion of irradiance on a tilted surface due to ground reflections. Any of the inputs may be DataFrames or scalars.

Parameters

surface_tilt : numeric

 Surface tilt angles in decimal degrees. Tilt must be ≥ 0 and ≤ 180 . The tilt angle is defined as degrees from horizontal (e.g. surface facing up = 0, surface facing horizon = 90).

ghi : numeric

 Global horizontal irradiance. [W/m²]

albedo : numeric, default 0.25

 Ground reflectance, typically 0.1-0.4 for surfaces on Earth (land), may increase over snow, ice, etc. May also be known as the reflection coefficient. Must be ≥ 0 and ≤ 1 . Will be overridden if surface_type is supplied.

surface_type: None or string, default None

 If not None, overrides albedo. String can be one of 'urban', 'grass', 'fresh grass', 'snow', 'fresh snow', 'asphalt', 'concrete', 'aluminum', 'copper', 'fresh steel', 'dirty steel', 'sea'.

Returns

grounddiffuse : numeric

 Ground reflected irradiance. [W/m²]

References

.. [1] Loutzenhiser P.G. et. al. "Empirical validation of models to compute solar irradiance on inclined surfaces for building energy simulation" 2007, Solar Energy vol. 81. pp. 254-267.

The calculation is the last term of equations 3, 4, 7, 8, 10, 11, and 12.

.. [2] albedos from:

<http://files.pvsyst.com/help/albedo.htm>

and

<http://en.wikipedia.org/wiki/Albedo>

and

[https://doi.org/10.1175/1520-0469\(1972\)029<0959:AOTSS>2.0.CO;2](https://doi.org/10.1175/1520-0469(1972)029<0959:AOTSS>2.0.CO;2)

""

if surface_type is not None:

 albedo = SURFACE_ALBEDOS[surface_type]

 diffuse_irrad = ghi * albedo * (1 - np.cos(np.radians(surface_tilt))) * 0.5

try:

 diffuse_irrad.name = 'diffuse_ground'

except AttributeError:

 pass

return diffuse_irrad

```
#####  
pvlib.irradiance.haydavies(surface_tilt, surface_azimuth, dhi, dni, dni_extra, solar_zenith=None,  
solar_azimuth=None, projection_ratio=None, return_components=False)  
  
#####  
def beam_component(surface_tilt, surface_azimuth, solar_zenith, solar_azimuth,  
dni):  
    """  
    Calculates the beam component of the plane of array irradiance.  
  
    Parameters  
    -----  
    surface_tilt : numeric  
        Panel tilt from horizontal.  
    surface_azimuth : numeric  
        Panel azimuth from north.  
    solar_zenith : numeric  
        Solar zenith angle.  
    solar_azimuth : numeric  
        Solar azimuth angle.  
    dni : numeric  
        Direct Normal Irradiance  
  
    Returns  
    -----  
    beam : numeric  
        Beam component  
    """  
    beam = dni * aoi_projection(surface_tilt, surface_azimuth,  
                                solar_zenith, solar_azimuth)  
    beam = np.maximum(beam, 0)  
  
    return beam
```

ANEXO B – BIBLIOTECA SCIPY E SUAS PRINCIPAIS FUNÇÕES UTILIZADAS

```
scipy.optimize.curve_fit(f, xdata, ydata, p0=None, sigma=None, absolute_sigma=False,  
check_finite=None, bounds=(-inf, inf), method=None, jac=None, *, full_output=False,  
nan_policy=None, **kwargs)
```

Use non-linear least squares to fit a function, f, to data.

```
scipy.signal.hilbert(x, N=None, axis=-1)[source]
```

Compute the analytic signal, using the Hilbert transform.

```
scipy.stats.skew(a, axis=0, bias=True, nan_policy='propagate', *, keepdims=False)
```

Compute the sample skewness of a data set.

```
scipy.stats.kurtosis(a, axis=0, fisher=True, bias=True, nan_policy='propagate', *, keepdims=False)
```

Compute the kurtosis (Fisher or Pearson) of a dataset.

```
scipy.stats.variation(a, axis=0, nan_policy='propagate', ddof=0, *, keepdims=False)
```

Compute the coefficient of variation.