**主变风险预测及基于主变状态的电网实时风险分析接入规范**

**V1.0**

**西安交通大学**

目 录

[1. 接口说明 3](#_Toc515974498)

[1.1 变压器风险预测(风险评估) 5](#_Toc515974499)

[1.2变压器故障后电网潮流快速估算及实时风险分析（直流潮流法） 6](#_Toc515974500)

[1.3变压器故障后电网潮流快速估算及实时风险分析（PQ分解法） 6](#_Toc515974501)

[1.4变压器故障后负荷削减及运行优化策略(负荷消减法) 7](#_Toc515974502)

[2. 输入参数说明 8](#_Toc515974503)

[2.1变压器风险预测 8](#_Toc515974504)

[2.2电网风险评估直流潮流法 9](#_Toc515974505)

[2.3电网风险评估PQ分解法 11](#_Toc515974506)

[2.4电网负荷削减及运行优化策略算法 12](#_Toc515974507)

[3. 输出参数说明 13](#_Toc515974508)

[3.1变压器风险预测结果 13](#_Toc515974509)

[3.2电网风险评估直流潮流法 13](#_Toc515974510)

[3.3电网风险评估PQ法结果 14](#_Toc515974511)

[3.4电网风险评估负荷消减结果 16](#_Toc515974512)

# 接口说明

所有算法采用统一调用方式：

UniParameteruniParameter = new UniParameter();

//TODO 设置uniParameter  
UniSolveruniSolver = new UniSolver();  
UniResultuniResult = uniSolver.solve(uniParameter, "COOL");

统一输入参数对象：

public class UniParameter {  
private IFhnlpgBaseiFhnlpgBase;  
 private IFhnlpgInitialiFhnlpgInitial;  
 private IFhnlpgOnLoadiFhnlpgOnLoad;  
 private IFhnlpgTRiseiFhnlpgTRise;  
 private IFhnlpgResistanceiFhnlpgResistance;  
 private IFhnlpgRatioiFhnlpgRatio;  
 private IFhnlpgLimitiFhnlpgLimit;  
 private IFhnlpgCooliFhnlpgCool;  
 private IFhnlpgTStatusiFhnlpgTStatus;  
 private IDwfxpgDWiDwfxpgDW;  
 private IDwfxpgPQiDwfxpgPQ;

privateIFXPG iFxpg;

}

统一输出对象：

public class UniResult {  
public interrcode= 0; //0：成功；-1:失败  
 public String errmsg; //失败原因  
 public String mode; //算法缩写：HST,ZCFHNLPG等  
 public OCooloCool;  
 public OHstoHst;  
 public OZcfhnlpgoZcfhnlpg;  
 public OCqfhnlpgoCqfhnlpg;  
 public ODqfhnlpgoDqfhnlpg;  
 public OFhztoFhzt;  
 public OZLCL oZlcl;  
 public OFHXJ oFhxj;  
 public OPQ oPq;

publicOFXPG oFxpg;

}

## 1.1 变压器风险预测(风险评估)

**1）算法模型**

|  |  |  |
| --- | --- | --- |
| 算法名称 | 变压器风险预测算法 |  |
| 输入数据 | 基本要求：   1. 变压器设备信息   变压器容量、运行年限、特别设备标识和负荷量等   1. 变压器状态评价信息   各组件评分值、整体评分值和评价结果   1. 变压器故障诊断结果   变压器高级综合诊断结论 |  |
| 输出数据 | 1. 变压器风险等级 2. 变压器风险因素 |  |

**2）算法接口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 类名 | UniSolver | | |
| 接口名 | Solve（） | | |
| 接口原型 | | UniResultuniResult = uniSolver.solve(uniParameter, "FXPG"); | | |
| 参数列表 | | 说明 | 参数类型 | 备注 |
| iFxpg | | 风险评估输入 | 对象 | 参加2.1 |
| 返回值 | | 说明 | 参数类型 | 备注 |
| oFxpg | | 变压器风险等级及说明 | 对象 | 详见3.1 |

## 1.2变压器故障后电网潮流快速估算及实时风险分析（直流潮流法）

**1）算法模型**

|  |  |
| --- | --- |
| 算法名称 | 变压器故障后电网潮流快速估算及实时风险分析算法 |
| 输入数据 | 基本要求：   1. 某变压器故障后的电网数据   节点标识、电压、功率，线路电阻、线路电抗、线路对地导纳、线路容量，变压器标准变比、变压器容量 |
| 输出数据 | 1. 电网潮流分布及实时风险结果 |

**2）算法接口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 类名 | UniSolver | | |
| 接口名 | Solve（） | | |
| 接口原型 | | UniResultuniResult = uniSolver.solve(uniParameter, "ZLCL"); | | |
| 参数列表 | | 说明 | 参数类型 | 备注 |
| iDwfxpgDW | | 电网参数 | 对象 | [详见2.](#_设备基本信息_EquipmentData)2 |
| 返回值 | | 说明 | 参数类型 | 备注 |
| oZlcl | | 电网潮流及电网风险 | 对象 | [详见3.2](#_设备基本信息_EquipmentData) |

## 1.3变压器故障后电网潮流快速估算及实时风险分析（PQ分解法）

**1）算法模型**

|  |  |
| --- | --- |
| 算法名称 | 变压器故障后电网潮流快速估算及实时风险分析算法 |
| 输入数据 | 基本要求：   1. 某变压器故障后的电网数据   节点标识、电压、功率，线路电阻、线路电抗、线路对地导纳、线路容量，变压器标准变比、变压器容量 |
| 输出数据 | 1. 电网潮流分布及实时风险结果 |

**2）算法接口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 类名 | UniSolver | | |
| 接口名 | Solve（） | | |
| 接口原型 | | UniResultuniResult = uniSolver.solve(uniParameter, "PQ"); | | |
| 参数列表 | | 说明 | 参数类型 | 备注 |
| iDwfxpgPQ | | PQ分解法输入 | 对象 | 详见2.3 |
| 返回值 | | 说明 | 参数类型 | 备注 |
| oPq | | PQ分解法结果 |  | [详见3.3](#_设备基本信息_EquipmentData) |

## 1.4变压器故障后负荷削减及运行优化策略(负荷消减法)

**1）算法模型**

|  |  |
| --- | --- |
| 算法名称 | 变压器故障后负荷削减计算及运行优化措施 |
| 输入数据 | 1. 某变压器故障后的电网数据   节点标识、节点负荷功率，节点发电功率、支路起始节点、终止节点支路标识、支路容量 |
| 输出数据 | 1. 变压器故障后负荷削减结果及运行优化措施 |

**2）算法接口**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口名称 | 类名 | UniSolver | | |
| 接口名 | Solve（） | | |
| 接口原型 | | UniResultuniResult = uniSolver.solve(uniParameter, "FHXJ"); | | |
| 参数列表 | | 说明 | 参数类型 | 备注 |
| iDwfxpgDW | | 负荷消减法输入完全同直流潮流法 | 对象 | 参见2.2 |
| 返回值 | | 说明 | 参数类型 | 备注 |
| oFhxj | | 负荷削减量及优化建议 | 对象 | 详见3.4 |

# 输入参数说明

## 2.1变压器风险预测

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **构造函数** | public class IFXPG {  private int score = 80; //健康评分  private String scoreMsg = ""; //状态评价的风险因素描述  private String diagnoseResult = ""; //诊断结果：异常状态描述  private double t = 10; //运行年限（年）  private double V = 180; //变压器容量（MVA）  private int x = 0; //设备标识  private double I = 800; //高压侧电流  }  public IFXPG(intscore, double t, double V, intx, double I) {  this.score= score;  this.t = t;  this.V= V; this.x= x; this.I= I; }  public void setScoreMsg(String scoreMsg) { this.scoreMsg= scoreMsg; }  public void setDiagnoseResult(String diagnoseResult) { this.diagnoseResult= diagnoseResult; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 健康评分 | Score | Y |  |  |
| 运行年限（年） | T | Y |  |  |
| 变压器容量（MVA） | V | Y |  |  |
| 设备标识 | X | Y |  |  |
| 高压侧电流 | I | Y |  |  |
| 状态评价的风险因素描述 | scoreMsg | Y |  |  |
| 诊断结果：异常状态描述 | diagnoseResult | Y |  |  |

## 2.2电网风险评估直流潮流法

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **构造函数** | public IDwfxpgDW(ArrayList<CBus> bus, ArrayList<CLoad> load, ArrayList<CGenerator> generator, ArrayList<CLine> line) { Bus = bus; Load = load; Generator = generator; Line = line; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 电网节点 | Bus | Y |  |  |
| 电网中有功负荷 | Load | Y |  |  |
| 电网中有功发电量 | Generator | Y |  |  |
| 电网支路数据 | Line | Y |  |  |
| **构造函数** | public CBus(intid, inttype) { this.id = id; this.type= type; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 节点标识 | **id** | Y |  | 编号必须从1开始，依次递增 |
| 节点类型 | type | Y |  | PQ节点为1，PV节点为2，平衡节点最后输入为3 |
| **构造函数** | public CLoad(intid, double pl) { this.id = id;  this.pl = pl; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 节点标识 | **id** | Y |  |  |
| 节点有功负荷 | **pl** | Y |  | 标幺值 |
| **构造函数** | public CGenerator(intid, double pg) { this.id = id;  this.pg = pg; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 节点标识 | **id** | Y |  |  |
| 节点有功发电量 | **pg** | Y |  | 标幺值 |
| **构造函数** | public CLine(intsid, inteid, intsn, double resistance, double capacity) { this.sid= sid; this.eid= eid; this.sn = sn; this.resistance= resistance; this.capacity= capacity; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 支路起始节点标识 | **sid** | Y |  | 对应bus的id |
| 支路终点节点标识 | **eid** | Y |  |  |
| 支路标识 | **sn** | Y |  | 线路为1，变压器支路为0 |
| 支路电抗 | **resistance** | Y |  | 标幺值 |
| 支路容量 | capacity | Y |  | 标幺值 |

## 2.3电网风险评估PQ分解法

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **构造函数** | public IDwfxpgPQ(ArrayList<CBus> bus, ArrayList<CLine> line) { Bus = bus; Line = line; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 电网节点数量 | Bus | Y |  |  |
| 电网支路（线路）数 | Line | Y |  |  |
| **构造函数** | public CBus(intid, inttype, double U, double a, double P, double Q) { this.id = id; this.type= type; this.U= U; this.a= a; this.P= P; this.Q= Q; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 节点标识 | **Id** | Y |  |  |
| 节点类型 | Type | Y |  | PQ为1，PV为2，平衡节点为3 |
| 节点电压 | U | Y |  | 标幺值 |
| 节点相角 | a | Y |  | 标幺值 |
| 节点有功功率 | P | Y |  | 负荷为负，发电量为正，标幺值 |
| 节点无功功率 | Q | Y |  | 容性为正，感性为负，标幺值 |
| **构造函数** | public CLine(intsid, inteid, double Rij, double Xij, double B0, double RT, double XT, double KT, double W, double capacity) { this.sid= sid; this.eid= eid; this.Rij= Rij; this.Xij= Xij; this.B0 = B0; this.RT= RT; this.XT= XT; this.KT= KT; this.W= W; this.capacity= capacity; } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 支路起始节点标识 | **Sid** | Y |  |  |
| 支路终点节点标识 | **Eid** | Y |  |  |
| 线路电阻 | **Rij** | Y |  | 标幺值，变压器支路此项为0 |
| 电抗 | **Xij** | Y |  | 标幺值，变压器支路此项为0 |
| 支路对地导纳 | **BO** | Y |  | 标幺值 |
| 变压器支路电阻 | **RT** | Y |  | 标幺值，若为线路支路此项为0 |
| 变压器支路电抗 | **XT** | Y |  | 标幺值，若为线路支路此项为0 |
| 变压器标准变比 | **KT** | Y |  | 标幺值 |
| 折算标志 | **W** | Y |  | 变压器等效支路阻抗折算高压侧为0，低压侧为1 |
| 支路容量 | **capacity** | Y |  | 标幺值 |

## 2.4电网负荷削减及运行优化策略算法

参见2.2电网风险评估直流潮流法

# 输出参数说明

## 3.1变压器风险预测结果

|  |  |  |
| --- | --- | --- |
| **类** | public class OFXPG { public String riskLevel;  public String riskDisp;  public String scoreMsg; //状态评价的风险因素描述 public String diagnoseResult; //诊断结果：异常状态描述 } | |
| **属性名称** | **代码值** | **备注** |
| 风险等级 | riskLevel |  |
| 风险来源 | riskDisp |  |
| 状态评价的风险因素描述 | scoreMsg | 对应输入同名变量 |
| 异常状态描述 | diagnoseResult | 对应输入同名变量 |

## 3.2电网风险评估直流潮流法

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **类** | public class OZLCL {  public ArrayList<CLinePIJ>PIJ;  public ArrayList<LineMsg>lineMsgs;  public ArrayList<BusMsg>busMsgs;  } | | | | |
| **属性名称** | **代码值** | | | **备注** | |
| 线路潮流 | PIJ | | |  | |
| 线路风险 | lineMsgs | | |  | |
| 节点相角 | busMsgs | | |  | |
| **构造函数** | public class CLinePIJ {  public int sid;  public int eid;  public double pij;   public CLinePIJ(int sid, int eid, double pij) {  this.sid = sid;  this.eid = eid;  this.pij = pij;  }   public int getSid() {  return sid;  }   public int getEid() {  return eid;  }   public double getPij() {  return pij;  } } | | | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | | **备注** | |
| 支路起始节点标示 | **Sid** | Y |  | |  | |
| 支路结束节点标示 | **Eid** | Y |  | |  | |
| 潮流值 | **Pij** | Y |  | |  | |
| **构造函数** | public class LineMsg {  public int sid;  public int eid;  public String msg1;  public int status;   public LineMsg(int sid, int eid, String msg1, int status) {  this.sid = sid;  this.eid = eid;  this.msg1 = msg1;  this.status = status;  }   public int getSid() {  return sid;  }   public int getEid() {  return eid;  }   public String getMsg1() {  return msg1;  }   public int getStatus() {  return status;  } } | | | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | | **备注** | |
| 支路起始节点标示 | **Sid** | Y |  | |  | |
| 支路结束节点标示 | **Eid** | Y |  | |  | |
| 线路计算结果描述 | **Msg1** | Y |  | |  | |
| 状态标示 | status | Y |  | | 不同的msg1对应不同的status | |
| **构造函数** | public class BusMsg {  public int id;  public String msg1;  public int status;   public BusMsg(int id, String msg1, int status) {  this.id = id;  this.msg1 = msg1;  this.status = status;  }   public BusMsg(int id, String msg1) {  this.id = id;  this.msg1 = msg1;  }   public int getId() {  return id;  }   public String getMsg1() {  return msg1;  }   public int getStatus() {  return status;  } } | | | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | | **备注** | |
| 节点标示 | **Id** | Y |  | |  | |
| 节点计算结果描述 | **Msg1** | Y |  | |  | |
| 状态标示 | **Status** | Y |  | |  | |

## 3.3电网风险评估PQ法结果

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **类** | public class OPQ {  public int k; //迭代次数 public Vector U; //节点电压 public Vector a; //节点相角 public Vector P; //节点有功 public Vector Q; //节点无功 public Vector I; //起始节点 public Vector J; //终止节点 public Complex Sph;  public VectorComplex Sij; public VectorComplex Sji; public VectorComplex deltaSij; public VectorComplex S; public Complex sumdeltaS; public ArrayList<LineMsg>lineMsgs; public ArrayList<BusMsg>busMsgs;  } | | | | |
| **属性名称** | **代码值** | | | **备注** | |
| 迭代次数 | k | | | 若迭代次数k大于20,errcode=-1, errormsg=“PQ分解法无法收敛，请转入线性潮流估算方法的直流法” | |
| 节点电压 | U | | | 标幺值 | |
| 节点相角 | a | | | 标幺值 | |
| 各节点有功功率 | P | | | 标幺值 | |
| 各节点无功功率 | Q | | | 标幺值 | |
| 节点起始标示 | I | | |  | |
| 节点结束标示 | J | | |  | |
| 平衡节点的复功率 | Sph | | | 标幺值 | |
| 各支路功率 | Sij, Sji | | | 标幺值. Sij代表I-J和Sji代表J-I | |
| 各支路功率损耗 | deltaSij | | | 标幺值 | |
| 各节点复功率 | S | | | 标幺值 | |
| 电网总损耗 | sumdeltas | | | 标幺值 | |
| 重载支路负载率 | LineMsgs | | | 是否有越限风险 | |
| 节点电压情况 | BusMsgs | | | 是否有越限风险 | |
| **构造函数** | public class Complex {  public double a;  public double b;  public double getA() {  return a;  }   public double getB() {  return b;  }  } | | | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | | **备注** | |
| 复数实部 | **a** | Y |  | |  | |
| 复数虚部 | **b** | Y |  | |  | |
| **构造函数** | public class VectorComplex {  private Complex[] c;  private int size;  public Complex[] getC() {  return c;  }  public int getSize() {  return size;  }  public Complex get(int i) {  return c[i];  }    public Vector getReal() {  Vector v = new Vector(size);  for (int i = 0; i < size; i++) {  v.set(i, c[i].a);  }  return v;  }   public Vector getImag() {  Vector v = new Vector(size);  for (int i = 0; i < size; i++) {  v.set(i, c[i].b);  }  return v;  }  } | | | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | | **备注** | |
| 复数数组 | **c** | Y |  | |  | |
| 数组大小 | **Size** | Y |  | |  | |
| **构造函数** | public class Vector {  private double[] A;  private int size;  }  public double[] getA() {  return A; }  public int getSize() {  return size; }  public double get(int i) {  return A[i]; }  public ArrayList<Double> toArrayList() {  ArrayList<Double> arrayList = new  ArrayList<Double>();  for (int i = 0; i < size; i++) {  arrayList.add(A[i]);  }  return arrayList; } | | | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | | **备注** | |
| 向量数组 | **A** | Y |  | |  | |
| 数组大小 | **size** | Y |  | |  | |

## 3.4电网风险评估负荷消减结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **构造函数** | public class OFHXJ extends OZLCL {  public intstatus; public String msg1; public double f;  } | | | |
| **参数名称** | **代码值** | **必填** | **默认值** | **备注** |
| 负荷消减结论 | msg1 | Y |  |  |
| 负荷量 | f | Y |  |  |
| 状态标志 | status | Y |  | 当status=0时，父对象OZLCL没有数据 |