Here is a brief explanation of the **NPLRML** (Parallel Linearization ADMM Algorithm for Solving **N**onconvex **P**enalized **L**ogistic **R**egression under **M**anual **L**abeling) package.

**Note :** Before using this R package, please ensure that the following four dependent R packages are installed, “glmnet”，”Matrix”，”gtools”，”ggplot2”.

**The main function:**

NPLRML: Parallel Linearization ADMM Algorithm for Solving **N**onconvex **P**enalized **L**ogistic **R**egression under **M**anual **L**abeling.

**The input and output of the main function:**

#' @param X Matrix of predictors, of dimension (n\*p); each row is an observation.

#' @param S Manual labeling, when using real response variables, is equivalent to solving ordinary penalized logistic regression

#' @param D The number of local machines

#' @param Pen Regularization term, such as 'LASSO' or 'SCAD' or 'MCP'

#' @param lambda Parameter tuning or regularization term parameters.

#' @param m Number of manual labeling experts

#' @param mu Lagrange's quadratic augmented constant

#' @param eta Linearization parameters

#' @param max\_iter Maximum number of algorithm iterations

#' @param tol Error parameters for algorithm termination

#' @returns \item{beta}{Regression coefficient.}

#' @returns \item{obj\_values}{Optimization value of objective function.}

#' @returns \item{iterations}{number of iterations.}

#' @returns \item{time}{calculation time.}

**######### Numerical experiments**

**library(glmnet)**

**library(Matrix)**

**library(gtools)#生成迪利克雷随机数所需要**

**rdirichletbinomial <- function(n=1, m, alpha0, Pr) {**

**# 生成Dirichlet随机变量**

**p <- gtools::rdirichlet(n, alpha0 \* Pr)**

**# 从二项分布生成计数数据**

**S <- sapply(1:n, function(i) rbinom(1, size = m, prob = p[i, 1]))**

**# 返回结果**

**list(p = p, S = S)**

**} #生成迪利克雷随机数**

**pite2=function(X,x,ite) #计算最大特征值函数**

**{**

**x\_m=matrix(0,nrow=ncol(X),ncol=ite+1)**

**x\_m[,1]=x**

**k=0**

**repeat{**

**x\_u=t(X)\%\*\%(X\%\*\%x\_m[,k+1])**

**x\_u=x\_u/sqrt(sum(x\_u^2))**

**lam\_u= (sum((X\%\*\%x\_u)^2))**

**k=k+1**

**x\_m[,k+1]=x\_u**

**err\_a<-max(abs(x\_m[,k+1]-x\_m[,k]))**

**if((err\_a<0.001)|(k>ite-1)){break}**

**}**

**return(lam\_u)**

**}**

**set.seed(999)**

**n <- 200 # 样本量**

**p <- 8 # 变量维度**

**X <- matrix(rnorm(n \* p), n, p)**

**true\_beta <- matrix(c(3,1.5,0,0,2,0,0,0),ncol=1) # 稀疏系数**

**pp <- X \%\*\% true\_beta**

**prob <- 1 / (1 + exp(-pp))**

**S1 <- rbinom(n, 1, prob) # 二元标签**

**alpha0 = 500**

**m <- 1000**

**y = rbinom(n, 1, prob)**

**S2 = rep(0,n)**

**for (i in 1:n) {**

**S2[i] = rdirichletbinomial(n = 1, m, alpha0, c(prob[i], 1-prob[i]))$S**

**}**

**cv.fit <- cv.glmnet(X, y, family = "binomial", alpha = 1)**

**lambda\_opt <- cv.fit$lambda.min**

**# 使用最优的 lambda 值获取模型系数**

**coef\_opt <- coef(cv.fit, s = lambda\_opt) #lambda\_opt**

**coef\_opt**

**lambda1 = 2**

**eta <- 0.7\*pite2(X,x=rep(1,ncol(X)),ite=100)[1] #norm(t(X) \%\*\% X, "2") \* rho \* 1.1 # 线性化参数（tau > rho\*||X'X||）**

**result0 <- NPLRML(X, y, D=1,Pen = "LASSO", lambda1, m = 1 ,**

**mu=1, eta = eta , max\_iter=1000, tol=1e-3)**

**result0$beta**

**result0$time**

**result0$iterations**

**# Penalized logistic regression with manual labeling by a single expert**

**result1 <- NPLRML(X, S1, D=1,Pen = "LASSO", lambda1, m = 1 ,**

**mu=1, eta = eta , max\_iter=1000, tol=1e-3)**

**result1$beta**

**result1$time**

**result1$iterations**

**# Penalized logistic regression with manual labeling by multiple experts**

**lambda2 = 200**

**lambda2/eta**

**result2 <- NPLRML(X, S2, D=1,Pen = "SCAD", lambda2, m ,**

**mu=1, eta = eta , max\_iter=1000, tol=1e-3)**

**result2$beta**

**sum(true\_beta - result2$beta)**

**result2$time**

**result2$iterations**

**# ----------------------------**

**# 3. 结果可视化**

**# ----------------------------**

**# 绘制目标函数下降曲线**

**plot(result2$obj\_values, type='l', xlab="Iteration", ylab="Objective Value",**

**main="Objective Function Convergence")**

**# 比较估计的beta和真实beta**

**library(ggplot2)**

**df <- data.frame(**

**index = 1:p,**

**true = true\_beta,**

**estimated = result2$beta**

**)**

**ggplot(df, aes(x=index)) +**

**geom\_point(aes(y=true, color="True"), size=3) +**

**geom\_point(aes(y=estimated, color="Estimated"), size=3) +**

**labs(title="True vs Estimated Coefficients",**

**y="Coefficient Value", x="Variable Index") +**

**scale\_color\_manual(values=c("True"="red", "Estimated"="blue")) +**

**theme\_minimal()**