祝介东北京力高泰科技有限公司

使用 *R* 语言分析 *LI-6400* 和 *LI-6800* 光合仪的数据

目录

表格

插图

欢迎

本文纯属个人打发出差漫漫长夜的作品，并非正式资料，并未经过校对等流程，但所用代码皆经过本人和同事的验证，所引文献均真实存在。希望对不熟悉相关知识的童鞋们有所帮助。

前言

本文内容均来自个人对相关材料的理解，事实上，直到这本书的内容定型时，我的代码水平还处在初级水平，因此，如果你刚接触 R，不要把这本书当作规范，因为成型后我几乎没怎么改过，一是懒，二是忙。文中内容**并未经过权威认证，如有异议，请参考 R 软件及相关软件包的使用手册或相关模型的文献**。本文所使用代码均只做示例代码使用，在做实际分析时，请结合自己的实验数据及模型使用条件做相应的调整。本文内容仅针对 LI-6800 与 LI-6400XT 光合仪数据，尤其是 RACiR™ 部分，仅针对 LI-6800 进行分析，且目前市面上仅 LI-6800 光合仪可完成此项数据的测量。如您有 LI-6400XT 或 LI-6800 使用操作相关方面的问题请发送至：

zhujiedong@ecotek.com.cn

如有与本文数据分析相关的内容请发送至我个人邮箱（本文是业余爱好，内容虽与公司仪器相关，但并不是我领薪水的工作内容）：

zhujiedong@yeah.net

以便我们对相关内容做出修正。

当然，如果您有 github 帐号，优先欢迎在 github 提交：

github 地址[1]

gitee 也有同步版本

gitee 地址[2]

如有其他仪器问题或者仪器购买需求，请使用下面方式与我们联系：

- 北京力高泰科技有限公司

[1] https://github.com/zhujiedong/photoanalysis

[2] http://zhu_jie_dong.gitee.io/photoanalysis/

- 网址：http://www.ecotek.com.cn
- 电话：010-64093960/66001653
- 电子邮件：info@ecotek.com.cn
- 地址：北京市西城区西直门南大街 2 号成铭大厦 A 座 22F

关于作者

祝介东，北京力高泰科技有限公司[3] 工程师，本文内容为我在售后服务时所住的酒店所作（遍布国内大多数省份），本为我打发出差住酒店漫漫长夜的业余爱好，**此部分内容并非公司对我的职位所要求的内容，也并非我公司提供的服务内容，因此属于个人作品，仅供参考，我所在公司对此文内容不负任何责任**。
文中代码得到了公司技术部同事：刘美玲、徐粒、焦阳、王昭、张启尧、张云飞等的测试，其中焦阳贡献了 `jiptest` 包中基础荧光参数计算的代码，在此一并感谢。

文中内容如对您有帮助，我很欣慰。但本人水平有限，错误与疏漏之处还请谅解，欢迎讨论，欢迎提出宝贵意见及建议。本人或单位其他售后人员仅有提供相关仪器操作或故障解决的义务，处理数据和分析数据并非我们的本职工作，如有相关需还请多参考本文内容或网上相关资料，不足之处还请谅解。

[3] http://www.ecotek.com.cn/

版权

本文旨在对广大 **LI-6400XT** 和 **LI-6800** 光合仪的用户提供一个数据分析的参考，本人或所在公司并未从中获取任何利益，内容错误疏漏之处，欢迎指正。本人保留一切权利，禁止一切将本文内容用于商业用途的行为，禁止商业公司使用。

1

R 软件与 Rstudio

1.1 R 软件

R 语言的官方网站是 R[1] ，与 R 语言有关的网站还有 CRAN（镜像），其主站网址是：CRAN[2]。

1.2 Rstudio

尽管 R 的功能十分强大，但是其作为一个命令行工具 [3]，在实际使用中尤为不便，因此，一款集成的开发环境十分有必要。Rstudio[4]是一款专门针对 R 开发的一个集成环境，同时也支持其他多种语言，用户界面十分友好，支持代码高亮，拼写提示，作图展示等功能，因此我们推荐使用 Rstudio 对我们的数据进行分析。

R 软件和 Rstudio 的安装十分方便，我们按照各自系统的安装方式安装即可，安装后界面及各区功能如下：

本文的主要内容并非介绍这些软件的功能，因此，关于软件的操作部分请仔细阅读相关资料，网络上有大量的免费资源及教程，有需要的同学可自行搜索。

[1] http://www.r-project.org

[2] http://www.cran.r-project.org

[3] 实际上在 Windows 系统下安装的时候提供一个十分简陋的 GUI。

[4] https://www.rstudio.com/products/RStudio/

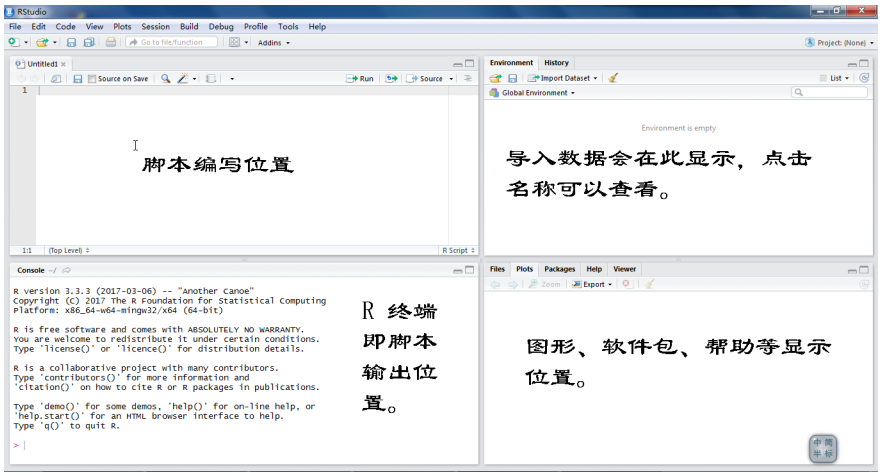


图 1.1: Rstudio 界面及功能

注：R 与 Rstudio 均非我公司产品，而且均免费或者有免费版本，因此请勿邮件或电话索要此两款软件。

2

批量处理光合测定数据

对于多数人来讲，一个季节用光合仪测量的数据文件至少是两位数的，处理起来非常不方便，针对这个问题，简单写了一个批量读取 LI-6400 和 LI-6800 原始数据的包 (因为现有的容易实现的读取 excel 格式的包还不支持 6800 和 6400 这种形式的公式计算)[1]，使用非常简单，同时也适合处理未关闭数据文件而导致的无法生成 excel 格式的数据时的问题。

2.1 安装

暂时只有我的 github repo 中的版本：

```
devtools::install_github("zhujiedong/readphoto")
```

2.2 6400 数据整合

基本参数如下：

[1]特别注意，原始数据可以用文本编辑器打开，但为了方便使用这个软件包，准确输入与行号相关的参数，建议您使用带行号显示的软件，例如 windows 下的 notepad++

表 2.1: LI-6400 批量整合数据

	files	Obs	HHMMSS	FTime	EBal.	Photo	Cond	Ci
1	aci	1	10:55:14	483.0	0	6.990	0.0831	251.0
4	aci	2	10:57:35	623.5	0	5.160	0.0853	192.0
7	aci	3	10:59:55	763.5	0	3.140	0.0881	136.0
10	aci	4	11:02:26	914.5	0	0.910	0.0927	81.9
13	aci	5	11:04:46	1055.0	0	-0.167	0.0966	52.7
16	aci	6	11:07:23	1211.5	0	5.240	0.1010	305.0
19	aci	7	11:09:43	1352.0	0	6.610	0.1040	284.0
22	aci	8	11:12:04	1492.5	0	9.280	0.1050	438.0
25	aci	9	11:14:24	1633.0	0	10.200	0.1020	616.0
28	aci	10	11:16:44	1772.5	0	10.500	0.0943	795.0
31	aci	11	11:19:49	1958.0	0	10.700	0.0853	970.0
34	aci	12	11:22:09	2097.5	0	11.100	0.0812	1150.0
41	aq	2	10:12:52	737.5	0	6.450	0.0700	239.0
44	aq	3	10:15:12	878.0	0	6.450	0.0684	235.0
47	aq	4	10:17:32	1017.5	0	5.960	0.0655	241.0

```
library(readphoto)
df64  <- read_bat_6400("./data/6400", header_line = 17, data_start = 27)
```

数据输出如下所示 (仅显示前 8 列数据)：

如果想另存为 csv 格式：

```
write.csv(df64, "./combined.csv")
```

header_line 表示你数据表头所在行，data_start 表示你数据起始行，Obs = 1 时所在行，不含表头。这个也要确认好了，不同的测量不能放在一起（当然一般不会出现这种情况，同一台仪器，处理数据当然是希望 aci 和 aci 在一

起，lrc 和 lrc 在一起，调查测量和调查测量在一起），不同的测量可能起始行不同，这样就会报错，特别需要注意的是，目前路径写法仅支持 “/” 分隔，不能使用 “\“ 作为分隔。例如在数据放在 D 盘的 6400 文件夹下，那么写法应为”d:/6400”, 不能为 **“d:\6400”**, 尽管后者对 R 是合法的，主要是因为我要区分你不同数据来源的文件是哪个，也即下文提到的 `df$files` 列。

其中，数据的来源在表格中第一列，叫做 files，是数据来源的文件名（即你起的名字）, 例如本例中你看到的 aci 是我之前数据里面 aci 响应曲线的数据。

这些数据可以用于后文相关的分析中，尤其是像 `fitacis` 这样的函数，因为本质上他们都是符合 `tidyverse` 样式的数据。

2.3 LI-6800 数据整合

此部分内容保留，但不建议再使用：

~~相比 6400，6800 参数要少，导入需要两行命令即可：~~

```
library(readphoto)
df <- read_bat_6800("./data/6800", data_start = 56)
```

~~其中就两个参数，第一个为 LI-6800 原始数据所在的文件夹，即文件夹内包含且仅包含 LI-6800 的原始数据数据。第三个参数即为你原始数据中测量数据的起始行，例如本例中使用数据测量值从 56 行开始。~~

~~数据输出如下所示 (仅显示前 8 列数据)：~~

~~其他注意事项见 LI-6400 数据整合部分~~

~~另外：今天添加了使用 regex 读取 LI-6800 原始文件的方法，这个只需要路径即可（2020-3-18），简单测试，~~

```
library(readphoto)
df2 <- read_regex68("./data/6800")
```

表 2.2: LI-6800 批量整合数据

files	obs	time	elapsed	date	plot	plant	TIME
aci	1	1513614617	0.0	20171218 10:30:16	1	1	1513614617
aci	2	1513614731	114.0	20171218 10:32:10	1	1	1513614731
aci	3	1513614886	269.0	20171218 10:34:45	1	1	1513614886
aci	4	1513615008	391.0	20171218 10:36:47	1	1	1513615008
aci	5	1513615127	510.0	20171218 10:38:46	1	1	1513615127
aci	6	1513615287	670.5	20171218 10:41:27	1	1	1513615287
aci	7	1513615410	793.0	20171218 10:43:29	1	1	1513615410
aci	8	1513615566	949.0	20171218 10:46:05	1	1	1513615566
aci	9	1513615701	1084.0	20171218 10:48:20	1	1	1513615701
aci	10	1513615831	1214.0	20171218 10:50:30	1	1	1513615831
aci	11	1513615940	1323.0	20171218 10:52:19	1	1	1513615940
aci	12	1513616064	1447.4	20171218 10:54:24	1	1	1513616064
lrc	1	1513612721	0.0	20171218 09:58:40	1	1	1513612721
lrc	2	1513612832	111.0	20171218 10:00:31	1	1	1513612832
lrc	3	1513612941	220.0	20171218 10:02:20	1	1	1513612941

```
## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]

## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]

## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]

## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

表 2.3: LI-6800 regex 方式批量整合数据

obs	time	elapsed	date	plot	plant	TIME	E
1	1513614617	0.0	2017-12-18 10:30:16	1	1	1513614617	0.0045726
2	1513614731	114.0	2017-12-18 10:32:10	1	1	1513614731	0.0047162
3	1513614886	269.0	2017-12-18 10:34:45	1	1	1513614886	0.0049729
4	1513615008	391.0	2017-12-18 10:36:47	1	1	1513615008	0.0052632
5	1513615127	510.0	2017-12-18 10:38:46	1	1	1513615127	0.0053619
6	1513615287	670.5	2017-12-18 10:41:27	1	1	1513615287	0.0053909
7	1513615410	793.0	2017-12-18 10:43:29	1	1	1513615410	0.0052242
8	1513615566	949.0	2017-12-18 10:46:05	1	1	1513615566	0.0052158
9	1513615701	1084.0	2017-12-18 10:48:20	1	1	1513615701	0.0053947
10	1513615831	1214.0	2017-12-18 10:50:30	1	1	1513615831	0.0054908
11	1513615940	1323.0	2017-12-18 10:52:19	1	1	1513615940	0.0053860
12	1513616064	1447.4	2017-12-18 10:54:24	1	1	1513616064	0.0052159
1	1513612721	0.0	2017-12-18 09:58:40	1	1	1513612721	0.0067439
2	1513612832	111.0	2017-12-18 10:00:31	1	1	1513612832	0.0065955
3	1513612941	220.0	2017-12-18 10:02:20	1	1	1513612941	0.0064493

2.4 重计算功能

如果只是将原始数据批量处理，那么在遇到叶片不能充满叶室的情况时会很麻烦，这里我们提供了重新计算功能，重新计算的参数包括光合速率，蒸腾速率，对水的气孔导度以及胞间二氧化碳浓度，当然计算他们所需要的一些中间数值也做了重计算，只不过多数时候我们用不到，我们仅看这四项。

2.4.1 LI-6400 数据重计算

参数的重计算函数为 `recomp_6400`, 其参数除了 `read_bat_6400` 所包含的参数外，还有叶面积 S, 以及叶片正反面的气孔比例，默认值分别为 6 和 0.5。

```
library(readphoto)
x1 <- read_bat_6400("./data/6400")
y1 <- recomp_6400("./data/6400", header_line = 17, data_start = 27, S = 6, K = 0.5)

x1$Photo - y1$Photo
```

```
##  [1] -0.0008873753  0.0026900500 -0.0012036469  0.0003483414  0.0006122641
##  [6] -0.0113872639 -0.0020986076  0.0004962787  0.0188727482 -0.0294595908
## [11] -0.0436611445 -0.0339083408  0.0046772165  0.0036653298  0.0030397988
## [16] -0.0105901673  0.0040624956  0.0017317049 -0.0073252290  0.0054977377
## [21]  0.0039736503  0.0021704065  0.0046772165  0.0036653298  0.0030397988
## [26] -0.0105901673  0.0040624956  0.0017317049 -0.0073252290  0.0054977377
## [31]  0.0039736503  0.0021704065
```

```
x1$Trmmol - y1$Trmmol
```

```
##  [1] -2.998596e-04  1.407338e-04  3.189451e-05 -4.586467e-04 -3.836822e-04
##  [6]  5.402725e-04 -2.344852e-04 -7.684772e-05  5.979599e-04 -6.534341e-04
## [11] -6.779145e-04  2.469749e-04  3.812201e-04  2.313957e-04  3.508312e-04
## [16] -2.794358e-04 -5.406530e-04  5.230606e-04 -9.183370e-04  7.638850e-04
## [21] -2.578893e-04  2.203045e-04  3.812201e-04  2.313957e-04  3.508312e-04
## [26] -2.794358e-04 -5.406530e-04  5.230606e-04 -9.183370e-04  7.638850e-04
## [31] -2.578893e-04  2.203045e-04
```

```
x1$Cond - y1$Cond
```

```
##  [1] -1.974217e-04 -3.594216e-04 -3.779119e-04 -3.806675e-04 -3.201411e-04
##  [6] -1.483324e-04 -7.803345e-04 -2.671018e-04  1.028977e-04 -3.966192e-04
## [11] -3.190769e-04 -2.314266e-04 -2.746300e-04  1.094050e-05 -4.584791e-05
```

```
## [16] -1.084094e-04 -1.827768e-04 -1.344969e-04 -1.714096e-04 -8.180257e-05
## [21] -4.687906e-05 -1.000424e-04 -2.746300e-04  1.094050e-05 -4.584791e-05
## [26] -1.084094e-04 -1.827768e-04 -1.344969e-04 -1.714096e-04 -8.180257e-05
## [31] -4.687906e-05 -1.000424e-04
```

```
x1$Ci-y1$Ci
```

```
##  [1]  0.434643936 -0.297820404 -0.308200950 -0.007847373 -0.035490198
##  [6]  0.433706824 -0.416734067 -0.052089770  0.147655545 -0.315797917
## [11] -0.271335987 -0.228968795  0.356519198  0.311487646  0.052196075
## [16]  0.557128947  0.058563406  0.300198435  0.052607786  0.339000061
## [21] -0.252622980 -0.494554616  0.356519198  0.311487646  0.052196075
## [26]  0.557128947  0.058563406  0.300198435  0.052607786  0.339000061
## [31] -0.252622980 -0.494554616
```

```
# half of original the area
y1 <- recomp_6400("./data/6400",  header_line = 17, data_start = 27, S = 3, K = 0.5)
y1$Photo/x1$Photo
```

```
##  [1] 2.000254 1.998957 2.000767 1.999234 2.007333 2.004346 2.000635 1.999893
##  [9] 1.996299 2.005611 2.008161 2.006110 1.998550 1.998863 1.998980 2.003671
## [17] 1.998391 1.999240 2.003866 1.995584 1.994199 2.010360 1.998550 1.998863
## [25] 1.998980 2.003671 1.998391 1.999240 2.003866 1.995584 1.994199 2.010360
```

```
# test with random area less than six
area <- 6 - runif(32, 1, 3)
y1 <- recomp_6400("./data/6400",  header_line = 17, data_start = 27, S = area, K = 0.
y1$Photo/x1$Photo
```

```
##  [1] 1.547708 1.335227 1.798815 1.622090 1.318467 1.625458 1.493146 1.619808
##  [9] 1.614637 1.628912 1.278236 1.300041 1.258628 1.335591 1.487851 1.812318
## [17] 1.238013 1.806044 1.572072 1.876383 1.329133 1.235421 1.329836 1.464199
## [25] 1.225033 1.397261 1.621279 1.268235 1.252590 1.899600 1.398159 1.539921
```

我们看到各个值之差非常小，因为我们使用的是相同的叶面积，理论上这两次读数的差异应为 0，但在实际计算过程中，有小数点位数的影响，所以某些值不完全为 0，但该差值足够小。我们将所有的数据叶面积减半后，二者比值也约等于 2.

2.4.2 LI-6800 数据重计算

2.4.2.1 原始数据的批量计算

此部分内容保留，但不建议继续使用 ~~参数的重计算函数为`recomp_6800`，其参数除了`read_bat_6800`所包含的参数外，还有叶面积 S，以及叶片正反面的气孔比例，默认值分别为 6 和 0.5。~~

```
library(readphoto)
x3 <- read_bat_6800("./data/6800")
y3 <- recomp_6800("./data/6800", S = 6, K = 0.5)
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
x3$A - y3$A
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
x3$E - y3$E
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
x3$gsw - y3$gsw
```

```
##  [1] -0.01231889 -0.01271539 -0.01330812 -0.01416506 -0.01491902 -0.01586479
##  [7] -0.01566857 -0.01545520 -0.01563183 -0.01537453 -0.01494554 -0.01433677
## [13] -0.02284560 -0.02291318 -0.02268118 -0.02266862 -0.02250392 -0.02213352
## [19] -0.02061493 -0.02006006 -0.01856400 -0.01678649 -0.01591079 -0.01457109
## [25] -0.01322393 -0.02345109 -0.02312573 -0.02227497
```

```
x3$Ci-y3$Ci
```

```
##  [1]  4.43187567  3.13678828  1.78620310  0.49515242 -0.08437000  3.51420195
##  [7]  3.66611259  5.56360761  6.76351008  7.70784028  8.45263315  9.25926640
## [13]  2.58978666  2.52629775  2.49784796  2.38619050  2.23225682  1.92200639
## [19]  1.29213044  0.92018214  0.76352452  0.55412066  0.20248257 -0.08219604
## [25] -0.26216626  2.36039345  2.40636618  2.54353742
```

```
# half of original the area
y3 <- recomp_6800("./data/6800", S = 3, K = 0.5)
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
y3$A/x3$A
```

```
##  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
# test with random area less than six
area <- 6 - runif(28, 1, 3)
y3 <- recomp_6800("./data/6800", S = area, K = 0.5)
```

```
## Warning: Missing column names filled in: 'X125' [125]
```

```
## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]

## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]

## Warning: Missing column names filled in: 'X125' [125]

## Warning: Duplicated column names deduplicated: 'TIME' => 'TIME_1' [69]
```

```
y3$A/x3$A
```

```
##  [1] 1.550904 1.440882 1.343701 1.968802 1.339638 1.433337 1.790952 1.381065
##  [9] 1.281886 1.908655 1.352130 1.300929 1.430593 1.275156 1.878866 1.248958
## [17] 1.257426 1.656351 1.636311 1.649412 1.634128 1.405141 1.864948 1.752261
## [25] 1.275639 1.756279 1.944309 1.391102
```

~~测量结果相比 6400 的数据，某些值差异略大，我仔细核对过公式，并无问题，可能仅仅是小数点后数据的~~
~~或者邮箱发送给我，便于我改正。~~

2.4.2.2 Excel 格式的重计算

偶然发现了 `XLConnect` 软件包的一个功能（以前知道这个软件包，但忽视了），那就是直接读取 LI-6800 Excel 格式的数据并重计算，我将其写成了函数，放在了我的 `readphoto` 软件包里，软件包的安装：

```
remotes::install_github("zhujiedong/readphoto")
```

当然，最近连我自己安装 github 的软件包都经常出问题，如果大家同样遇到问题，可以按照下面的方式安装：

```
remotes::install_git("https://gitee.com/zhu_jie_dong/readphoto")
```

其中：

- path 是 Excel 文件的路径；
- start_row 是数据开始的行号；
- S 为修改的叶面积，默认值为 6，如果叶面积无需更改，则使用默认的 NULL。如果使用 aperture 更改了面积，且叶片能够充满叶室，比方说是 2 cm^2，该值必须输入一个长度和测量值数量完全一致的向量，例如有 3 个测量值，我们输入 S 的长度则为 3，例如，一共有三个测量值，只有第一个叶片没充满叶室，面积为 1.5，其他的为 2，则输入方式为 `S = c(1.5, 2, 2)`。

我们直接使用下面的例子解释，导入的数据是 6 cm^2 的默认面积：

```
library(readphoto)

df6 <- xlconnect_read("./data/aci-xlc.xlsx")
df6$A
```

```
##  [1] 24.7381184 18.1379358 10.8055345  3.0239340 -0.9144044 26.9519572
##  [7] 27.5088717 40.9101889 50.1393342 55.3865984 58.0662751 59.3556428
```

将面积改为 3 cm^2

```
df3 <- xlconnect_read("./data/aci-xlc.xlsx", S = rep(3, 12))
df6$A/df3$A
```

```
##  [1] 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
```

模拟 12 个不同的叶面积，均值为 3，方差为 0.1

```
df_random <- xlconnect_read("./data/aci-xlc.xlsx", S = rnorm(12, 3, 0.1))
df6$A/df_random$A
```

```
##  [1] 0.4888643 0.5218187 0.4761483 0.4912074 0.4983397 0.5022426 0.5142848
##  [8] 0.4724804 0.4930526 0.5059951 0.4978962 0.5060074
```

2.4.2.3 批量处理 LI-6800 的数据

为了避免麻烦，不再建议前面 2.4.2的方式进行处理，其实基本的代码处理也很简单，例如下面演示的一些方法，供参考：

```
library(readphoto)
file_names <-
  list.files(path = "./data/extdata",
             full.names = TRUE,
             pattern = '*.xlsx')

data_list <- lapply(file_names, xlconnect_read)

## 整理好数据为正常格式并导出，例如导出 csv 文件

# 根据需要，若文件夹不存在可创建
#dir.create( "./data/extdata/csvdata")
short_names <-
  list.files(path = "./data/extdata", pattern = '*.xlsx')
short_name <- gsub(".xlsx", "", short_names)
csv_file <-
  paste0("./data/extdata/csvdata", "/", short_name, ".csv")
mapply(x = data_list, file = csv_file, write.csv)
```

```
# 若需要修改叶面积，则可以将叶面积重新另存为只有叶面积数据的文件
# 面积数据的文件仅包含 S 列，且文件名与待修改面积的文件名保持一致


# 批量读取叶面积的数据
# 为方便使用，我将面积存为了 csv 格式
# 根据需要，若文件夹不存在可创建
#dir.create( "./data/extdata/areacsv")
file_area <-
  list.files(path = "./data/extdata/area",
             full.names = TRUE,
             pattern = '*.csv')
area_list <- lapply(file_area, read.csv)

# mapply 使用多个 list 读取数据，注意将 list 命名为参数的名字，
# 以免出现使用位置的方式匹配参数，出现错误
data_list_csv <-
  mapply(
    path = file_names,
    S = area_list,
    xlconnect_read,
    start_row = 17,
    SIMPLIFY = F
  )
short_names_csv <-
  list.files(path = "./data/extdata", pattern = '*.xlsx')
short_name_csv <- gsub(".xlsx", "", short_names_csv)
csv_file_area <-
  paste0("./data/extdata/areacsv", "/", short_name_csv, ".csv")
mapply(x = data_list_csv, file = csv_file_area, write.csv)
```

```
## 如果要合并数据到一个文件内，可以在合并时，使用文件名作为
# 单独的一列，例如列名都叫做 from_file
# 例如使用我们刚刚修改叶面积的数据为例，使用刚刚导出的 csv 文件，节省代码


add_col_name <- function(path, file_name){
  df <- read.csv(path)
  df$from_file <- file_name
  df
  }

csv_data_path <-
  list.files(path = "./data/extdata/areacsv",
             full.names = TRUE,
             pattern = '*.csv')

data_combine <-
  mapply(
    path = csv_data_path,
    file_name = short_name_csv,
    add_col_name,
    SIMPLIFY = F
  )

# 或者使用 data.table::rbindlist(listOfDataFrames)
# 或者使用 dplyr::bind_rows(listOfDataFrames)

df_combined <- dplyr::bind_rows(data_combine)

# 检验数据
```

```
head(df_combined$from_file)
tail(df_combined$from_file)
```

3

CO_2 响应曲线的拟合

3.1 FvCB 模型

在 `plantecophys` 包中使用的模型为 Farquhar et al. (1980) 建立的 C3 植物模型 FvCB，其基于 C3 植物碳反应的三个阶段：

- 核酮糖-1,5-双磷酸羧化酶/加氧酶 (Rubisco) 的催化下, 核酮糖-1,5-双磷酸 (RuBP) 与 CO_2 发生羧化作用, 生成 3-磷酸甘油酸 (PGA)。
- 在腺苷三磷酸 (ATP) 和还原型烟酰胺腺嘌呤二核苷酸磷酸 (NADPH) 的作用下, PGA 被还原成磷酸丙糖 (TP)。每 6 个 TP 中有 1 个输出到细胞液中, 用于蔗糖或者淀粉的合成。
- 剩下的 5 个 TP 在 ATP 的作用下再生为 3 个 RuBP。一部分再生的 RuBP 在 Rubisco 的催化下被氧化成 PGA 和 2-磷酸乙醇酸, 2-磷酸乙醇酸在 ATP 的作用下形成 PGA, 并且释放 CO_2 (光呼吸)。

在光照下, C3 植物净光合速率 (A) 取决于 3 个同时存在的速率: RuBP 羧化速率 (Vc)、RuBP 氧化速率 (或光呼吸速率, Vo) 和线粒体在光照下的呼吸速率 (或明呼吸速率, Rd; 此名为了与暗呼吸速率对应和区分)。RuBP 氧化过程中每结合 1 mol O_2 就会释放 0.5 mol CO_2 。因此, 净光合速率 A 的计算为：

$$A = V_c - 0.5V_o - R_d \tag{3.1}$$

线粒体 Rd 不同于暗呼吸速率 (Rn)。Rn 是叶片在黑暗中的线粒体呼吸速

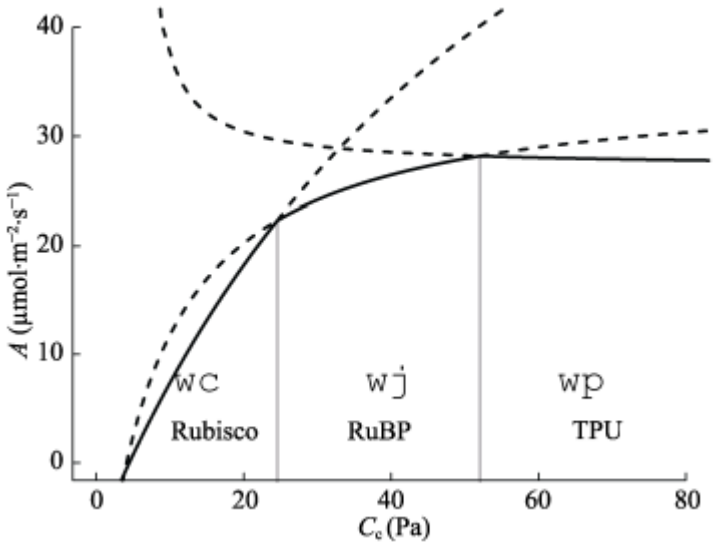


图 3.1: 光合速率的不同的限制阶段

率, 随着光照的增加, 线粒体呼吸速率下降。因此 Rd < Rn 在黑暗条件下测定的叶片 CO_2 交换速率即 Rn , 但是 Rd 的测定比较困难, 因为光照条件下 Rd 与 Vc、Vo 同时存在。Hikosaka et al. (2006) 总结了几种测定 Rd 的方法，式 (3.1) 又可表达为：

$$A = V_c\ (1\ -0.5\alpha)\ R_d \tag{3.2}$$

式 (3.2) 中 α 为氧化速率和羧化速率的比值，由 Rubisco 动力学常数确定：

$$\begin{aligned} \alpha &= \frac{V_O}{V_C} \\ &= \frac{O}{C_c} \times \frac{V_{omax} K_c}{V_{cmax} K_o} \\ &= \frac{O}{C_c} \times \frac{1}{S_{\frac{c}{o}}} \end{aligned} \tag{3.3}$$

式 (3.3) 中，Cc 和 O 分别为叶绿体部位 CO_2 和 O_2 浓度。Cc 和 O 通常以气体摩尔分数 ($\mu mol \bullet mol^{-1}$) 或分压 (Pa) 表示, 但光合过程是在叶绿体的液相基质中发生的, 用分压表示更加恰当。Kc 与 Ko 为 Rubisco 羧化 (氧

化) 的米氏常数, 代表了羧化 (氧化) 速率达到最大羧化 (氧化) 速率一半时的 CO_2 和 O_2 浓度。是 Rubisco 特异性因子, $S_{\frac{c}{o}}$ 表示 Rubisco 对 CO_2 和 O_2 的偏好程度。

当 A = Rd , 即 RuBP 羧化的 CO_2 吸收速率刚好等于 RuBP 氧化的 CO_2 释放速率 (Vc = 2Vo，羧化时 CO_2 提供的为 CO) 时, $\alpha = 0.5$。此时叶绿体的 CO_2 浓度就是叶绿体 CO_2 光合补偿点, 标记为 Γ^*。即：

$$\Gamma^* = \frac{0.5O}{S_{\frac{c}{o}}} \tag{3.4}$$

由公式 (3.3)和公式 (3.4)可得：

$$\alpha = \frac{2\Gamma^*}{C_c} \tag{3.5}$$

代入公式 (3.2) 得到：

$$A = V_c \left(1 - \frac{\Gamma^*}{C_c}\right) R_d \tag{3.6}$$

在 Cc 浓度很低的时候，RuBP 供应充足 (图 3.1 Rubisco 阶段)，Vc 等于 Rubisco 所能支持的羧化速率 w_c，：

$$w_c = \frac{V_{cmax}\, C_c}{C_c + K_c \left(1 + \frac{O}{K_o}\right)} \tag{3.7}$$

随着 Cc 浓度的增加，Rubisco 支持的羧化速率超过了 RuBP 供应速率, Vc 受 RuBP 再生速率的限制（图 3.1 RuBP 阶段），此时 Vc 由 RuBP 的再生速率限制，而 RuBP 又由电子传递速率（J）决定，故：

$$w_j = \frac{J\, C_c}{4C_c + 8\Gamma^*} \tag{3.8}$$

当 Cc 浓度很高，光合磷酸化超过了淀粉和蔗糖的合成速率的时候，Vc 受到 TP 利用速率（Vp）的限制（图 3.1 TPU 阶段），一般情况下，

$$w_p = \frac{3V_pC_c}{C_c - \Gamma^*} \tag{3.9}$$

最终，C3 植物叶片的光合速率 A 由 w_c、w_j、w_p 的最小者决定（图 3.1 实现部分），当 $c > \Gamma^*$ 时：

$$A = min\{w_c, w_{j,} w_p\}(1 - \frac{\Gamma^*}{C_c}) - R_d \tag{3.10}$$

3.2 CO_2 响应曲线测量的注意事项

尽管上文对其分段性做了数学上的解释，相比来讲，不是那么通俗易懂，根据 Haworth et al. (2018) 文章中的内容，我们后面两小节的内容对其进行概述：

3.2.1 分段性

与光响应曲线不同，A/Ci 曲线是分段的，这也增加了其复杂性，在其最初阶段，CO_2 浓度较低，在此阶段，Rubisco 更多的与 O_2 结合，因此，即使是较小浓度的 CO_2 的增加，也会显著提高羧化速率，我们将此阶段称之为 CO_2 (Wullschleger (1993)) 限制或 Rubisco 限制 (Long and Bernacchi (2003))。净光合速率 A 与 Ci 在此阶段斜率比较陡峭，实践中往往利用计算该斜率来计算 V_{cmax}。

在较高的 CO_2 浓度下，曲线斜率开始便的平缓，作为底物的 CO_2 已经不在是限制因素，随着羧化速率达到最大，RUBP 的量成为了其限制因素，而 RUBP 的再生速率受最大电子传递速率 J_{max} 的限制。此时曲线的弯曲点由 V_{cmax} 限制转变为 J_{max}，许多研究中将饱和光下和 CO_2 浓度下测量的光合速率称之为做大光合速率 (Heath et al. (2005))。而另一些研究中将最大光合速率定义为外界 CO_2 时，在饱和光强下达到的最大光合速率 (Marshall and Biscoe (1980))。这些术语上的差别需要注意。

在之后，有可能继续观测到磷酸盐限制 RUBP 再生的情况，导致光合速率的下降。因为此时大量的磷酸丙糖与正磷酸盐结合，导致 ATP 合成受限制 (Ellsworth et al. (2015))。这就是 TPU 限制阶段。

3.2.2 测量注意事项

尽管您的操作是严格按照说明书操作的，但说明书是指仪器的正确操作方式，无法对所有测量都采用相同设置，要获得好的测量结果，有更多的因素需要注意：

使用 LI-6400 或 LI-6800 测量 A/Ci 曲线的过程也就是控制叶室或参比室气体浓度变化的过程，**只要诱导的时间足够**，气孔会在相应设置的环境条件下开到足够大，这样 Ci 会随 Ca 而变化，一般来讲二者的比例为 0.7，但也可能在 0.5~0.7 间变化。

一般来讲，测量参数是在温度为 25 °C 时获得，但实际测量过程中，因为外界温度过高或过低等无法控制叶室温度到 25 °C，这其实并非严重的问题，因为这可以通过数学上的方法将计算参数标准化为 25 °C 时的结果。所以，在测量时只需控制叶室温度稳定即可（通常为 20 °C ~ 30 °C 之间）。此外就是控制恒定的 VPD 及一个饱和光强。另外就是需要注意，开始测量之前，必须在外界环境的 CO_2 浓度下诱导足够长的时间，使光合速率达到稳定，一般需要 20 ~ 30 min。对于没有稳定的光保护机制的植物，一般不建议在 50 ppm 或更低的浓度下进行设置，此时饱和光强和建议的温度下，植物没有足够的底物进行光合作用，这样会因为光化学反应的降低发生氧化性损伤。Centritto et al. (2003) 研究表明，长时间的在 50 ppm 下诱导气孔打开到最大时，可以观测到最大的气孔导度（非标准方式测量）。

对于存在干旱胁迫的测量，由于干旱会导致气孔关闭（Lauteri et al. (2014)）, 此时没有足够多的 CO_2 进入气孔，此时的测量是没有意义的，可在 50 ppm 诱导 1 h 等待气孔完全打开再快速升高 Ca 的值来进行测量（Centritto et al. (2003)）。该方法对于 V_{cmax} 不受影响而 J_{max} 降低的情况适用（Aganchich et al. (2009)）。但在某些情况下，气孔关闭速度太快，无法完成整个 A/Ci 曲线过程 (Haworth2017)（需要考虑 LI-6800 RACiR）。更重要的是，如果想采用拟合方式求 gm，那么气孔必须完全打开使叶片对 CO_2 吸收的限制降到最低。对于灌溉情况良好的植物或者土壤水分情况比较好的植物，气孔不对高于外界浓度的 Ca 的升高而响应 (Haworth et al. (2015))，这可能需要更多的测量点或延长测量点的时间间隔来提高曲线的分辨率。另外，测量点的数量也要根据研究而改变，例如重点测量 Vcamx 时，50 ~ 300 ppm 的数据

点要多一些，而如果研究对象是土壤磷酸盐对植物生理的限制，那么 1600 ~ 2000 ppm 的数据点要适当增多。

一个更精确的了解植物生理指标的方法是将 A/Ci 曲线改为 A/Cc 曲线，但这需要了解 gm 数据。因为 Cc 通过如下方式计算：

$$C_c = C_i - \frac{A}{gm} \tag{3.11}$$

对于 gm 的计算，比较易操作的有几种：采用光合荧光联合测量的方式计算求得。当然也可以采用曲线拟合的方式，或者 Yin et al. (2009) 使用的方式，在低氧气体下，采用不同的光照水平求得。

此外，测量气体交换非常重要的误差来源就是气体的扩散，因为测量时，多数时间内外界气体浓度要高于叶室内的气体浓度，那么即使使用密封性非常好的材料，由外界高 CO_2 浓度气体向叶室低 CO_2 浓度气体的扩散无法避免，尤其是在连续长时间测量时该效应尤为明显，因此需要经常更换叶室垫圈。具体可以通过一些方法来校正（Flexas J and A (2007)，rodeghiero2007major），但如果采用 LI-6800 测量这将不是问题，它采用的叶室增加技术并根据测量的漏气情况对结果自动修正。

3.3 plantecophys 软件包

Remko A. Duursma 在 2015 年发表了一篇文章 Duursma (2015)，plantecophys 是其开发的一个 R 包工具集，用于对叶片气体交换数据进行分析和建模。实现了如下功能：

- CO_2 响应曲线 (A-Ci curves) 的拟合、作图及模拟。
- 不同气孔导度模型。
- 根据 Cowan-Farquhar 的假设估算最适的气孔导度。
- 耦合气体交换模型的实现。
- 基于 Ci 模拟 C4 光合。
- RHtoVPD：常用单位的转换（相对湿度、水汽压亏缺、露点温度）。

各参数的基本用法请参考后文内容，或官方帮助文档:

plantecophys[1]。

3.4 LI-6400XT CO_2 响应曲线的拟合

LI-6400XT CO_2 响应曲线的拟合需要借助 plantecophys 的 fitaci 实现，fitaci 函数为根据 FvCB 模型对 A-Ci 曲线的测量数据进行拟合，并估算 J_{max}、V_{cmax}、R_d 及他们的标准差，并根据 Medlyn et al. (2002) 的方法考虑了温度的影响。

3.4.1 fitaci 函数介绍

fitaci 的用法如下[2]：

[1] https://cran.r-project.org/web/packages/plantecophys/plantecophys.pdf
[2] 仅针对 C3 植物

```
fitaci(data, varnames = list(ALEAF = "Photo",
  Tleaf = "Tleaf", Ci = "Ci", PPFD = "PARi",
  Rd = "Rd"), Tcorrect = TRUE, Patm = 100,
  citransition = NULL, quiet = FALSE,
  startValgrid = TRUE, fitmethod =
  c("default", "bilinear", "onepoint"),
  algorithm = "default", fitTPU = FALSE,
  useRd = FALSE, PPFD = NULL, Tleaf = NULL,
  alpha = 0.24, theta = 0.85, gmeso = NULL,
  EaV = 82620.87, EdVC = 0, delsC = 645.1013,
  EaJ = 39676.89, EdVJ = 2e+05,
  delsJ = 641.3615, GammaStar = NULL,
  Km = NULL, id = NULL, ...)

## S3 method for class 'acifit'
plot(x, what = c("data", "model", "none"),
     xlim = NULL, ylim = NULL, whichA = c(
       "Ac", "Aj", "Amin", "Ap"), add = FALSE,
pch = 19, addzeroline = TRUE, addlegend =
  !add, legendbty = "o",transitionpoint = TRUE,
linecols = c("black", "blue", "red"), lwd = c(1,
2), ...)
```

主要参数注释：

- data：需要分析的数据，必须为 data.frame[3]。格式。
- varnames：数据的表头，此处函数默认的表头为 LI-6400 的表头，分析 LI-6400 的数据时可以不用填写，直接使用默认的参数即可[4]。
- Tcorrect：如果为 TRUE，那么 J_{max}、V_{cmax} 的结果为温度校正结果，若 Tcorrect = FALSE，则为测量温度下的结果。

[3]具体参考 R 语言相关文档，为 R 语言最常用的数据格式

[4]在 R 中，使用参数的值为默认值时可以不用填写该参数，例如使用默认选项分析 LI-6400 数据时，可只填写 data 项，具体参考 R 的相关入门手册

- Patm：为外界大气压。
- citransition：参见详，若提供该选项，则 J_{max}、V_{cmax} 的区域则分别拟合[5]。
- fitmethod：参见详解。
- fitTPU：是否拟合 TPU 限制，默认为 FALSE，参见详解。
- x：对于 plot.acifit，x 为 fitaci 返回的对象，简单理解为将 fitaci 函数拟合结果赋值给一个变量，此处 plot 函数实际上为 plot.acifit。
- what：利用基础做图工具，默认为对数据和模型进行作图。
- whichA：默认为对所有的光合速率进行作图（Aj=Jmax-limited (蓝色), Ac=Vcmax- limited (红色), Hyperbolic minimum (黑色)), TPU-limited rate (Ap, 如果模型有计算结果）。

其他参数请参考 FvCB 模型 Farquhar et al. (1980) 或查看 plantecophys 的帮助文档。

3.4.1.1 fitaci 函数详解

- 默认为非线性拟合，详见 Duursma (2015)。
- bilinear 方法使用两次线性拟合方法首先拟合 V_{cmax} 和 R_d，然后在拟合 J_{max}，过渡点的选择为对所有数据拟合最适的点，类似 Gu et al. (2010) 的方法。该方法的优势时无论如何，都会返回拟合结果，尤其是非线性拟合失败时使用该方法，但若默认方法失败时，需首先检查是否数据存在问题。两种拟合方法的结果有轻微的差别[6]。
- onepoint 参考 De Kauwe et al. (2016)。
- citransition 使用时，数据将被区分为 V_{cmax} 限制（Ci < citransition ）区域，以及 J_{max} 限制 (Ci > citransition) 区域。
- fitTPU：如果要计算 TPU，要设置 fitTPU = TRUE，并且 fittingmethod = “bilinear”。但需要注意，当 TPU 被计算时，没有 J_{max} 限制的点的存在是可能的。TPU 限制的发生是在 A 值不随 CO_2 的增加而增加时发生的，因此计算 TPU 是否有返回值，取决于测量数据是否有此情况出现。

[5]为 Rubisco 和 RuBP 限制的 Ci 转换点，物种间差异较大，可以通过预实验确定

[6]若默认拟合方法失败，数据也无问题，那么是非线性拟合初始值设定的原因

3.5 使用 plantecophys 拟合 LI-6400XT CO_2 响应曲线数据

3.5.1 数据的前处理

虽然 R 软件支持直接导入 xlsx 的数据，但因为 LI-6400XT 的数据记录文件内有其他空行或 remark 等内容，增加了处理代码的量，**故而推荐将其数据先整理为如表 3.1 样式，并另存为 csv 格式**[7]：

表 3.1: 推荐 LI-6400 整理后数据样式

Obs	HHMMSS	FTime	EBal.	Photo	Cond	Ci	Trmmol
1	15:46:59	271.5	0	14.2848912	0.2730691	286.39751	2.226126
2	15:48:26	358.0	0	10.6562220	0.2826303	217.32002	2.292845
3	15:49:54	446.0	0	6.4525814	0.2909460	150.67623	2.361704
4	15:51:26	538.5	0	1.7971569	0.3057164	85.82530	2.459459
5	15:52:54	626.5	0	-0.6575974	0.3150002	53.47985	2.515992
6	15:54:50	742.5	0	15.4296572	0.3255415	292.56161	2.579840

3.5.2 使用示例

`plantecophys` 并非 base 的安装包，首次使用需要从 `CRAN` 安装，可以使用图形界面安装，也可以直接用命令行安装[8]，推荐同时安装依赖。

```
install.packages("plantecophys", dependencies = TRUE)
```

[7] 即仅保留测量值，删除其他所有头文件、空行、remark 等信息

[8] 首次使用安装，更换电脑或者升级 R 软件后，如果没有拷贝 library，也需要运行安装命令

```
# 载入 plantecophys
library("plantecophys")

# 利用 read.csv 读取数据文件，
# 我的路径为当前工作路径的 data 文件夹内
aci <- read.csv("./data/aci.csv")

# 防止可能出现的 NA 值
aci <- subset(aci, Obs > 0)

# 不修改默认参数对数据进行拟合
acifit <- fitaci(aci)
# 查看拟合结果的参数名称，方便导出数据使用
attributes(acifit)
```

```
## $names
##  [1] "df"             "pars"           "nlsfit"         "Tcorrect"
##  [5] "Photosyn"       "Ci"             "Ci_transition"  "Ci_transition2"
##  [9] "Rd_measured"    "GammaStar"      "Km"             "kminput"
## [13] "gstarinput"     "fitmethod"      "citransition"   "gmeso"
## [17] "fitTPU"         "alphag"         "RMSE"           "runorder"
##
## $class
## [1] "acifit"
```

```
# 查看拟合结果
summary(acifit)
```

```
## Result of fitaci.
##
## Data and predictions:
```

```
##            Ci      Ameas     Amodel         Ac         Aj   Ap       Rd VPD
## 5    53.47985 -0.6575974 -0.5146882 -0.3552036  0.000000 1000 0.159449 1.5
## 4    85.82530  1.7971569  1.9292621  2.0888575  5.068534 1000 0.159449 1.5
## 3   150.67623  6.4525814  6.4176037  6.5777528 12.755502 1000 0.159449 1.5
## 2   217.32002 10.6562220 10.5354626 10.6965875 17.519644 1000 0.159449 1.5
## 1   286.39751 14.2848912 14.3365887 14.4993980 20.749310 1000 0.159449 1.5
## 6   292.56161 15.4296572 14.9749157 15.1383702 20.852616 1000 0.159449 1.5
## 7   292.96456 15.7134791 15.0564801 15.2200522 20.831098 1000 0.159449 1.5
## 8   450.64285 22.2659015 23.0115187 23.1975997 25.186939 3000 0.159449 1.5
## 9   622.03873 26.5135040 27.6485003 30.4281393 27.837462 3000 0.159449 1.5
## 10  992.08737 30.3898585 30.6300461 42.3998173 30.797660 3000 0.159449 1.5
## 11 1558.96968 33.6267056 32.6638021 54.9948264 32.828110 3000 0.159449 1.5
## 12 1756.16396 33.3152783 33.0981490 58.5844507 33.261965 3000 0.159449 1.5
##       Tleaf         Cc     PPFD Patm Ci_original
## 5  31.12332   53.47934 1800.490  100    53.47985
## 4  30.99093   85.82723 1800.558  100    85.82530
## 3  30.82872  150.68265 1800.140  100   150.67623
## 2  30.63983  217.33057 1800.524  100   217.32002
## 1  30.46890  286.41186 1800.701  100   286.39751
## 6  31.26338  292.57660 1799.923  100   292.56161
## 7  31.41866  292.97963 1799.975  100   292.96456
## 8  31.54122  450.66588 1799.826  100   450.64285
## 9  31.63493  622.06640 1799.578  100   622.03873
## 10 31.73910  992.11803 1800.055  100   992.08737
## 11 31.86938 1559.00238 1800.022  100  1558.96968
## 12 31.96654 1756.19709 1799.585  100  1756.16396
##
## Root mean squared error:  1.889701
##
## Estimated parameters:
##         Estimate Std. Error
## Vcmax  49.261616  1.5152405
## Jmax  126.620537  2.2816267
```

```
## Rd      0.159449  0.4001302
## Note: Vcmax, Jmax are at 25C, Rd is at measurement T.
##
## Curve was fit using method:  default
##
## Parameter settings:
## Patm = 100
##  alpha = 0.24
##  theta = 0.85
##  EaV = 82620.87
##  EdVC = 0
##  delsC = 645.1013
##  EaJ = 39676.89
##  EdVJ = 2e+05
##  delsJ = 641.3615
##
## Estimated from Tleaf (shown at mean Tleaf):
## GammaStar =  58.61138
## Km =  1223.279
```

```
acifit_linear <- fitaci(aci,  fitmethod = "bilinear", quiet = TRUE)
summary(acifit_linear)
```

```
## Result of fitaci.
##
## Data and predictions:
##            Ci      Ameas     Amodel         Ac        Aj   Ap        Rd VPD
## 5    53.47985 -0.6575974 -0.7389447 -0.3560483  0.000000 1000 0.3828608 1.5
## 4    85.82530  1.7971569  1.7108198  2.0938246  5.138366 1000 0.3828608 1.5
## 3   150.67623  6.4525814  6.2098476  6.5933940 12.931333 1000 0.3828608 1.5
## 2   217.32002 10.6562220 10.3375299 10.7220229 17.761317 1000 0.3828608 1.5
## 1   286.39751 14.2848912 14.1477696 14.5338762 21.035722 1000 0.3828608 1.5
```

```
## 6   292.56161 15.4296572 14.7876514 15.1743678 21.139657 1000 0.3828608 1.5
## 7   292.96456 15.7134791 14.8694171 15.2562440 21.117718 1000 0.3828608 1.5
## 8   450.64285 22.2659015 22.8464806 23.2527613 25.533374 3000 0.3828608 1.5
## 9   622.03873 26.5135040 27.8030690 30.5004944 28.220254 3000 0.3828608 1.5
## 10  992.08737 30.3898585 30.8295808 42.5006398 31.221072 3000 0.3828608 1.5
## 11 1558.96968 33.6267056 32.8913778 55.1255986 33.279305 3000 0.3828608 1.5
## 12 1756.16396 33.3152783 33.3316070 58.7237587 33.719013 3000 0.3828608 1.5
##       Tleaf         Cc     PPFD Patm Ci_original
## 5  31.12332   53.47911 1800.490  100    53.47985
## 4  30.99093   85.82701 1800.558  100    85.82530
## 3  30.82872  150.68244 1800.140  100   150.67623
## 2  30.63983  217.33037 1800.524  100   217.32002
## 1  30.46890  286.41167 1800.701  100   286.39751
## 6  31.26338  292.57641 1799.923  100   292.56161
## 7  31.41866  292.97944 1799.975  100   292.96456
## 8  31.54122  450.66572 1799.826  100   450.64285
## 9  31.63493  622.06656 1799.578  100   622.03873
## 10 31.73910  992.11823 1800.055  100   992.08737
## 11 31.86938 1559.00260 1800.022  100  1558.96968
## 12 31.96654 1756.19733 1799.585  100  1756.16396
##
## Root mean squared error:  2.013045
##
## Estimated parameters:
##          Estimate Std. Error
## Vcmax  49.3787547  3.4815555
## Jmax  128.5546403         NA
## Rd      0.3828608  0.4697008
## Note: Vcmax, Jmax are at 25C, Rd is at measurement T.
##
## Curve was fit using method:  bilinear
##
## Parameter settings:
```

```
## Patm = 100
##  alpha = 0.24
##  theta = 0.85
##  EaV = 82620.87
##  EdVC = 0
##  delsC = 645.1013
##  EaJ = 39676.89
##  EdVJ = 2e+05
##  delsJ = 641.3615
##
## Estimated from Tleaf (shown at mean Tleaf):
## GammaStar =  58.61138
## Km =  1223.279
```

```
# 仅查看拟合参数，比较两种拟合参数的差异
coef(acifit_linear)
```

```
##       Vcmax        Jmax          Rd
##  49.3787547 128.5546403   0.3828608
```

```
coef(acifit)
```

```
##      Vcmax       Jmax         Rd
##  49.261616 126.620537   0.159449
```

```
# 设置作图参数，图形的边距及分为 1 行两列输出图形
par(mar = c(4.5, 4.5, 2, 2))
par(mfrow = c(1, 2))
# 对两种拟合参数的结果作图，查看模型拟合是否正常
```

```
plot(acifit, addlegend = FALSE)
legend(x = 500, y = 10,
       legend = c(expression(paste(A[c])),
                  expression(paste(A[j])),
                  "Limiting rate"),
       lty = c(1, 1, 1),
       col =c("red", "blue", "black")
         )
mtext(" fitmethod = 'default' ")

plot(acifit_linear, addlegend = FALSE)
legend(x = 500, y = 10,
       legend = c(expression(paste(A[c])),
                    expression(paste(A[j])),
                    "Limiting rate"),
       lty = c(1, 1, 1),
       col =c("red", "blue", "black")
)
mtext("fitmethod = 'bilinear' ")
```

如果需要导出数据做他用，直接根据 attributes 中看到的名称，选择对应的数据导出即可，如果使用 Rstudio 的话，其自动补全的功能在选择数据上更方便。例如导出预测值和系数分别使用如下方式：

```
# 将模型拟合结果中 df（即计算数据）赋给 predictaci，
# 并用 write.csv 导出
predictaci <- acifit$df
write.csv(acifit$df, file = "acipredict.csv")
write.csv(coef(acifit), file = "coefaci.csv")
```

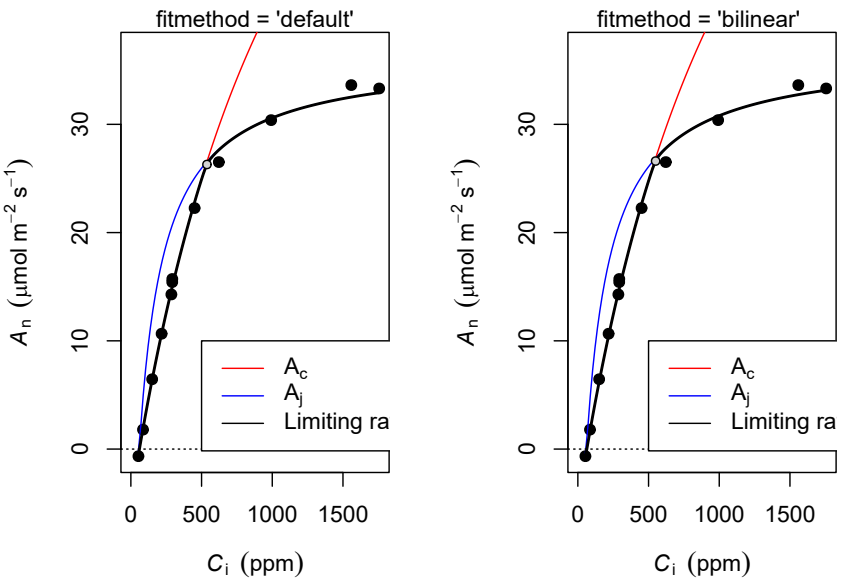


图 3.2: 光合速率的不同的限制阶段

需要注意的是，因为非线性拟合需要一个初始值，因此，使用默认方式（非线性拟合）的时候，会存在可能的拟合失败现象，此时可以使用 `fitmethod = "bilinear"`，二者结果略有差别。

3.5.2.1 fitmethod = "onepoint" 介绍

De Kauwe et al. (2016) 发表了关于 one point 方法计算 V_{cmax} 和 J_{max} 方法的文章，在 2017 年 11 月的更新中，plantecophys 增加了响应的 R 软件实现方法，该方法并非使用一个点计算 V_{cmax} 和 J_{max}，而是对数据集中的每一个点的值进行估计，使用的方法为逆向了光合作用方程。输出为对每个原始数据加入了 V_{cmax} 和 J_{max}，当然一如既往的可以使用温度校准的方法。**并不建议该方法应用于整个 ACi 曲线的数据，它的假设是在外部环境 CO_2 浓度和饱和光下，受到 Rubisco 羧化速率的限制而不是 RUBP 的限制。**

基于上面的描述，他们的模型如下：

$$\hat{V}_{cmax} = (A_{sat} + R_{day})\frac{C_i + K_m}{C_i - \Gamma^*} \tag{3.12}$$

其中：K_m 为米氏常数，其计算为：

$$K_m = K_c(1 + \frac{O_i}{K_o}) \tag{3.13}$$

未知参数均由文献中的方法进行计算，具体可参考 De Kauwe et al. (2016) 的原文, 但上述方法的缺陷为还要使用 ACi 曲线来估算 R_{day}，因此作者使用了 1.5% V_{cmax} 作为 R_{day}，因此公式 (3.12) 可变换为：

$$\hat{V}_{cmax} = A_{sat}(\frac{C_i + K_m}{C_i - \Gamma^*} - 0.015) \tag{3.14}$$

另一个重要的模型的假设为 J_{max} 与 V_{cmax} 是成比例的，J_{max} 的计算是通过 C_i transition point 来实现的，文章中的比值均值为 1.9，范围在 1.68 ~ 2.14 之间。

3.5.3 使用 'onepoint' 单独计算 V_{cmax} 和 J_{max}

目前我手头没有相应数据，仅有使用 LI-6400 测试 auto log 2 时的一个数据，我们用这个来示范该方法的使用：

```
one_data <- read.csv("./data/onepoint.csv")
knitr::kable(head(one_data), booktabs = TRUE,
             caption = 'onepoint 使用的数据')
```

数据如上所示，为同一个叶片连续记录数据，故所有的光合速率十分接近。

使用方法：

```
library(plantecophys)
```

表 3.2: onepoint 使用的数据

Obs	HHMMSS	FTime	EBal.	Photo	Cond	Ci	Trmmol	VpdL	CTleaf
1	14:12:43	347.0	0	10.519216	0.1369336	263.0905	1.963046	1.445782	25.89411
2	14:13:04	369.0	0	10.361122	0.1357092	264.1135	1.946022	1.445559	25.89542
3	14:13:25	373.0	0	10.207166	0.1342114	264.8348	1.925734	1.445717	25.89660
4	14:13:41	389.5	0	9.547416	0.1281947	269.6337	1.854272	1.454248	25.94020
5	14:16:16	540.0	0	10.288968	0.1376602	267.9198	2.009217	1.471788	26.01813
6	14:16:32	555.5	0	10.178603	0.1381995	269.5898	2.016100	1.471138	26.01657

表 3.3: onepoint 法计算的结果

aci_fit.Photo	aci_fit.Vcmax	aci_fit.Jmax
10.519216	47.06893	71.73759
10.361122	46.22091	70.51059
10.207166	45.44617	69.35838
9.547416	41.86217	64.24400
10.288968	45.09335	69.36954
10.178603	44.37360	68.41628

```
one_data <- subset(one_data, Obs > 0)
one_data$Rd <- 0.5
aci_fit <- fitaci(one_data, fitmethod = "onepoint")
```

需要注意，为保证结果的精确，如果不设定 Rd, 也即文献中的 Rday，模型是无法计算的，因此上面的示例中虚构了一个，实际操作用一般使用低氧的 ACi 测量计算。

3.5.4 多条 CO_2 响应曲线的拟合

fitacis 函数实际上是 fitaci 函数的扩展，方便一次拟合多条曲线[9]。函数的参数如下：

```
fitacis(data, group, fitmethod = c("default",
        "bilinear"),progressbar = TRUE,
        quiet = FALSE, id = NULL, ...)

## S3 method for class 'acifits'
plot(x, how = c("manyplots", "oneplot"),
     highlight = NULL, ylim = NULL,
     xlim = NULL, add = FALSE, what = c("model",
     "data", "none"), ...)
```

主要参数详解：

实际上 fitacis 与 fitaci 模型算法完全一致，只不过增加了一个 group 参数，用于区分不同测量的数据，具体请参考举例内容。

3.5.4.1 fitacis 函数应用举例

下文代码根据 plantecophys 中的示例代码修改，进行演示，原代码请参考其帮助文档。

```
library(plantecophys)
# 只提取前 10 个不同测量的数据，节省时间进行举例
manyacidat2 <- droplevels(manyacidat[manyacidat$Curve %in%
                                       levels(manyacidat$Curve)[1:10],])

# 对多条曲线进行拟合，使用 bilinear 方法，
```

[9]需要注意，此时 fitmethod 一般推荐使用 bilinear。

```
# 仅仅因为其比非线性拟合节省时间
fits <- fitacis(manyacidat2, group = "Curve", fitmethod="bilinear", quiet = TRUE)

# 拟合结果为 list, 我们可以只提取第一个的拟合结果
fits[[1]]
```

```
## Result of fitaci.
##
## Data and predictions:
##            Ci      Ameas     Amodel         Ac        Aj   Ap        Rd VPD
## 2    53.23129 -0.4401082  0.1014381  0.9601119  2.548123 1000 0.8586158 1.5
## 3    79.47367  2.4824630  2.1937702  3.0526198  7.036734 1000 0.8586158 1.5
## 4   116.74688  5.4531712  4.9419337  5.8011511 11.392394 1000 0.8586158 1.5
## 5   188.00801  9.7099879  9.5705964 10.4310194 16.447715 1000 0.8586158 1.5
## 6   278.44662 14.8225766 14.4261545 15.2897486 19.977583 1000 0.8586158 1.5
## 7   343.03259 17.7982155 17.4602014 18.3289218 21.639847 1000 0.8586158 1.5
## 1   344.72152 17.9244012 17.3165146 18.1849643 21.534276 1000 0.8586158 1.5
## 14  344.74839 16.7933747 17.6853306 18.5545917 21.774261 1000 0.8586158 1.5
## 8   588.08078 23.8925326 24.1309683 27.0327638 25.020148 3000 0.8586158 1.5
## 9   833.25547 26.5674409 25.7783026 33.0856065 26.647921 3000 0.8586158 1.5
## 10 1136.99222 25.9787890 26.8108335 38.1296944 27.676768 3000 0.8586158 1.5
## 11 1436.86370 26.6110657 27.5409345 42.0628463 28.405453 3000 0.8586158 1.5
## 12 1536.46772 27.4018784 27.7965881 43.3773689 28.660781 3000 0.8586158 1.5
## 13 1731.76400 28.6752069 28.0952804 45.2475932 28.959041 3000 0.8586158 1.5
##       Tleaf         Cc     PPFD Patm Ci_original
## 2  24.55873   53.23139 1799.959  100    53.23129
## 3  24.58292   79.47586 1799.590  100    79.47367
## 4  24.71278  116.75183 1799.819  100   116.74688
## 5  24.73687  188.01759 1800.371  100   188.00801
## 6  24.67508  278.46106 1800.233  100   278.44662
## 7  24.76596  343.05006 1799.575  100   343.03259
## 1  24.51593  344.73886 1800.356  100   344.72152
```

```
## 14 24.94098  344.76609 1799.964  100   344.74839
## 8  24.83785  588.10494 1799.477  100   588.08078
## 9  24.91185  833.28127 1799.969  100   833.25547
## 10 24.87314 1137.01906 1799.525  100  1136.99222
## 11 24.95914 1436.89126 1799.615  100  1436.86370
## 12 25.04542 1536.49554 1799.784  100  1536.46772
## 13 25.07566 1731.79212 1799.160  100  1731.76400
##
## Root mean squared error:  2.196037
##
## Estimated parameters:
##          Estimate Std. Error
## Vcmax  65.0009909  1.3720635
## Jmax  131.7980133         NA
## Rd      0.8586158  0.2876248
## Note: Vcmax, Jmax are at 25C, Rd is at measurement T.
##
## Curve was fit using method:  bilinear
##
## Parameter settings:
## Patm = 100
##  alpha = 0.24
##  theta = 0.85
##  EaV = 82620.87
##  EdVC = 0
##  delsC = 645.1013
##  EaJ = 39676.89
##  EdVJ = 2e+05
##  delsJ = 641.3615
##
## Estimated from Tleaf (shown at mean Tleaf):
## GammaStar =  42.31453
## Km =  698.2084
```

```
# 使用 sapply 提取拟合结果的 RMSE(均方根误差)
rmses <- sapply(fits, "[[", "RMSE")
plot(rmses, type='h', ylab="RMSE", xlab="Curve nr")
```

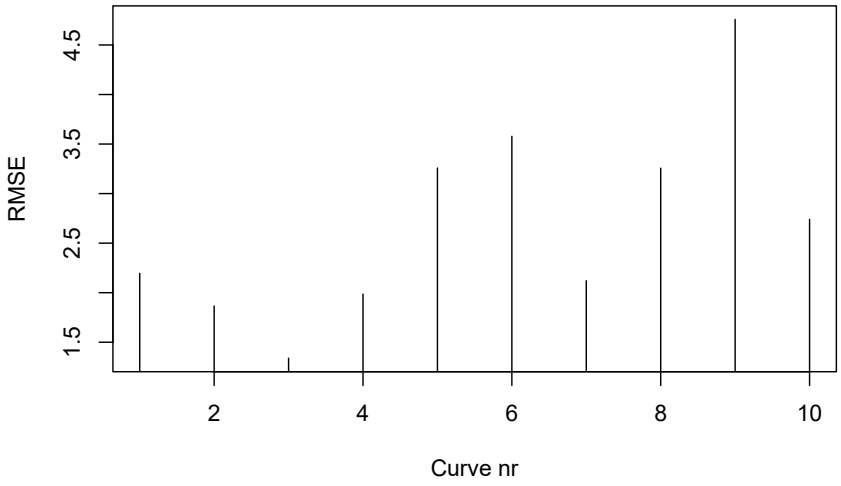


图 3.3: fitacis 作图结果

```
# 对最差的拟合结果进行作图
plot(fits[[which.max(rmses)]])
```

可以看出，`fitaci` 和 `fitacis` 用法基本一致，各行代码均已经注释，更详细用法请参函数考帮助。

3.5.5 `findCiTransition` 函数

计算 CiTransition 的函数，第一点为 Ac & Aj，第二点为 Aj & Ap，并且仅在计算 TPU 的前提下才会有第二点出现。

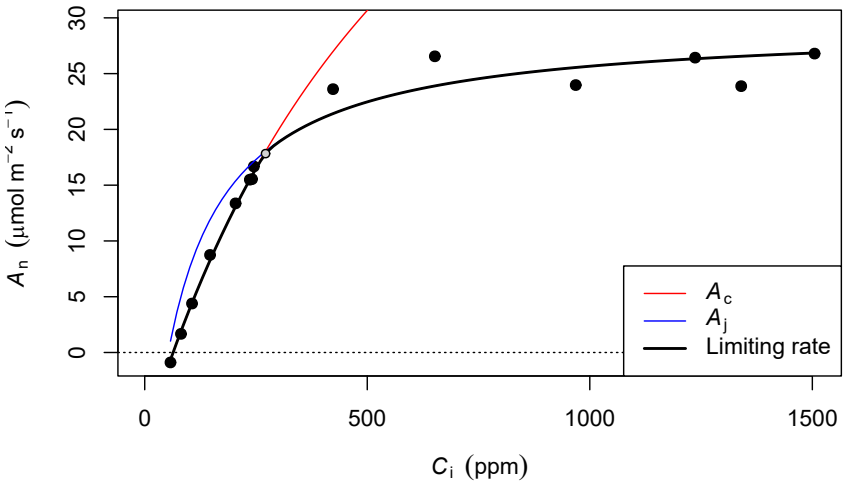


图 3.4: fitacis 作图结果

```
findCiTransition(object, ...)
```

参数使用，object 为 fitaci 函数对象，或者整个的 `Photosyn` 函数。... 为使用 `Photosyn` 时可传递的参数。

3.6 C4 植物光合

之前的部分模型全部为关于 C3 植物的拟合，而 Caemmerer (2000) 的方法，则是针对 C4 植物的 A-Ci 曲线的实现。

```
AciC4(Ci, PPFD = 1500, Tleaf = 25, VPMAX25 = 120,
      JMAX25 = 400, Vcmax = 60, Vpr = 80,
      alpha = 0, gbs = 0.003, O2 = 210,
      x = 0.4, THETA = 0.7, Q10 = 2.3,
      RD0 = 1, RTEMP = 25, TBELOW = 0,
      DAYRESP = 1, Q10F = 2, FRM = 0.5, ...)
```

参数详解

- Ci：胞间二氧化碳浓度 ($\mu mol \cdot m^{-2} \cdot s^{-1}$)。
- PPFD：光合光量子通量密度 ($\mu mol \cdot m^{-2} \cdot s^{-1}$)。
- Tleaf：叶片温度 (℃)。
- VPMAX25：PEP 羧化最大速率 ($\mu mol \cdot m^{-2} \cdot s^{-1}$)。
- JMAX25：最大电子传递速率 (℃))。
- Vcmax：最大羧化速率 ($\mu mol \cdot m^{-2} \cdot s^{-1}$)。
- Vpr：PEP 再生 ($\mu mol \cdot m^{-2} \cdot s^{-1}$)。
- alpha：维管束鞘细胞中 PSII 活性的比例。
- gbs：维管束鞘导度 ($mol \cdot m^{-2} \cdot s^{-1}$)。
- O2：叶肉细胞氧气浓度。
- x：电子传递的分配因子。
- THETA：曲角参数。
- Q10：Michaelis-Menten 系数中依赖于温度的参数。
- RD0：基温下的呼吸 ($mol \cdot m^{-2} \cdot s^{-1}$)。
- RTEMP：呼吸的基温 (℃)
- TBELOW：此温度以下呼吸为 0。
- DAYRESP：明呼吸和暗呼吸的比值。

- Q10F：呼吸依赖于温度的参数。
- FRM：明呼吸中为叶肉呼吸的比例。

以上参数均来自 Caemmerer (2000)，括号中的参数值均为默认值，具体应用时请按照实际情况修改。

3.6.1 C4 植物光合速率的计算

```
# 模拟 C4 植物的 Ci 值，计算光合速率并作图
library(plantecophys)
aci <- AciC4(Ci=seq(5,600, length=101))
with(aci, plot(Ci, ALEAF, type='l', ylim=c(0,max(ALEAF))))
```

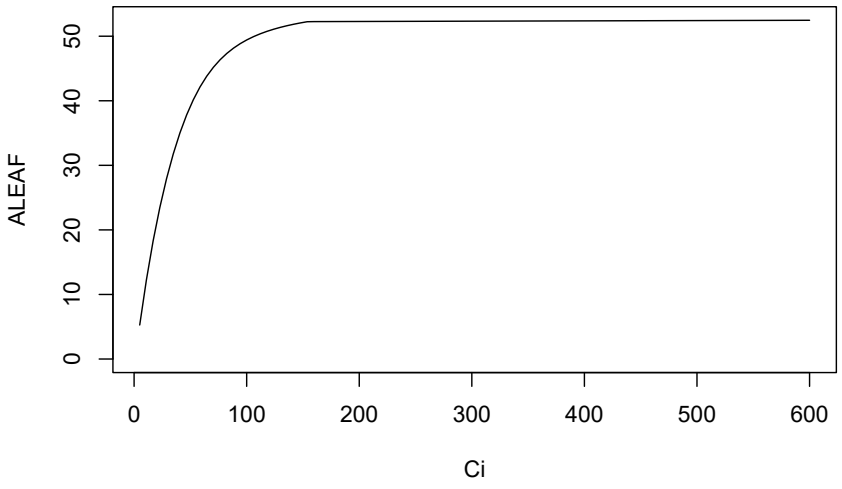


图 3.5: C4 植物 A-Ci 作图

4

气孔导度模型的拟合

气孔导度模型的拟合是通过 `fitBB` 来实现的，可以拟合三个 **Ball-Berry 类型的气孔导度模型**，共有下面几个参数：

- 气孔导度 (gs),
- 光合 (A),
- 外界 CO_2 浓度 (Ca)
- 水气压亏缺 (VPD).

其三个模型的简介如下：

4.1 BallBerry 模型

Ball et al. (1987) 发表的文章中的模型：

$$g_s = g0 + g1(\frac{Ah_r}{C_a}) \tag{4.1}$$

其中 A 为净光合速率，g_0 和 g_1 为拟合参数，h_r 为叶片表面的相对湿度，C_a 为叶片处 CO_2 浓度。

4.2 BBLeuning 模型

Leuning (1995) 发表的文章中的模型：

$$g_s = g_0 + g_1(\frac{A}{(C_a - \Gamma)(1 + \frac{D}{D_0})}) \tag{4.2}$$

其中 Γ 为 CO_2 补偿点，g_0、g_1 和 D_0 为拟合参数。

4.3 BBOptiFull 模型

Medlyn et al. (2011) 发表的文章中的模型：

$$g_s^* \approx g_0 + g_1(1 + \frac{g_1}{D})\frac{A}{C_a} \tag{4.3}$$

额外的参数 g_k 来自于 Duursma et al. (2013)

$$g_s = g_0 + 1.6(1 + \frac{g_1}{D}(1 - g_k))\frac{A}{C_a} \tag{4.4}$$

4.4 fitBB 函数

```
fitBB(data, varnames = list(
  ALEAF = "A", GS = "gsw", VPD = "VPDleaf",
  Ca ="CO2_s", RH = "RHcham"),
  gsmodel = c("BBOpti", "BBLeuning", "BallBerry",
              "BBOptiFull"), fitg0 = FALSE)
```

参数的意义：

- data：待分析的数据文件。
- varnames：注意，函数默认数据为 6400 格式，因此 6800 的数据文件要安装上文的参数修改。相对湿度只有在使用 BallBerry 时才需要输入。
- gsmodel：上述三个模型之一。
- fitg0：默认不计算 g0，若需要，改为 TRUE。

代码示例：

```
library(plantecophys)

aci <- read.csv("./data/aci.csv")
aci <- subset(aci, Obs > 0)
fitBB(aci, varnames = list(ALEAF = "Photo", GS = "Cond", VPD = "VpdL",
  Ca = "CO2S", RH = "RH_S"), gsmodel = "BBOpti", fitg0 = TRUE)
```

```
## Result of fitBB.
## Model :  BBOpti
## Both g0 and g1 were estimated.
##
## Coefficients:
## g0  g1
## 0.326 -0.992
##
## For more details of the fit, look at summary(myfit$fit)
## To return coefficients, do coef(myfit).
## (where myfit is the name of the object returned by fitBB)
```

4.5 fitBBs 函数

如果我们有多个物种的数据，那么使用 fitBBs 则可以快速拟合多条曲线的数据。我们先整合两次的数据，然后再查看运行结果：

```
aci01 <- read.csv("./data/aci01.csv")
aci01 <- subset(aci01, Obs > 0)
multiBB <- data.frame(
  A = c(aci$Photo, aci01$Photo),
  GS = c(aci$Cond, aci01$Cond),
  CO2S = c(aci$CO2S, aci01$CO2S),
  VPD = c(aci$VpdL, aci01$VpdL),
  RH = c(aci$RH_S, aci01$RH_S),
  species = c(rep("species1", length(aci$Photo)),
  rep("species2", length(aci01$Photo)))
)

mod2 <- fitBBs(multiBB, group = "species",
               varnames = list(
               ALEAF = "A", GS = "GS", VPD = "VPD",
               Ca ="CO2S", RH = "RH"),
               gsmodel = "BallBerry", fitg0 = TRUE)
```

```
## RH provided in % converted to relative units.
## RH provided in % converted to relative units.
```

```
coef(mod2)
```

```
##      group         g0         g1
```

```
## 1 species1 0.32638852 -0.1734554
## 2 species2 0.05158725 -0.0218842
```

5

光合最优气孔导度耦合模型

Cowan and Farquhar (1977) 关于最优气孔导度的描述概括如下：最优气孔行为理论认为气孔的最优化行为就是在某一时间段内，最大化光合碳固定的同时最小化蒸腾作用，也就是说，对于一定的水分消耗，最大化光合碳固定。即使得：

$$A - \lambda E \tag{5.1}$$

有最大化，其中 λ 是临界水分利用效率，也即植物损耗单位水分的 C 生产量，单位一般为 $mol\, CO_2 \cdot mol^{-1} H_2O$，可通过光合速率和蒸腾速率计算。

5.1 FARAO 函数

FARAO 函数用于找到最大化 $A - \lambda E$ 的 ** Ci ** 值。

```
FARAO(lambda = 0.002, Ca = 400, VPD = 1,
      photo = c("BOTH", "VCMAX","JMAX"),
      energybalance = FALSE, C4 = FALSE,
      Tair = 25, Wind = 2, Wleaf = 0.02,
      StomatalRatio = 1, LeafAbs = 0.86, ...)

FARAO2(lambda = 0.002, Ca = 400,
       energybalance = FALSE, ...)
```

其参数同 `fitaci` 和 `Photosyn`，在此不多做介绍，可参考 (Cowan and Farquhar, 1977, Buckley et al. (2014), Medlyn et al. (2011))。

6

光合气孔导度耦合模型

Duursma (2015) 对于气体交换耦合模型的简述如下 [1]：

- 对于 FvCB 模型有：

$$A_n = min(A_c, A_j) - R_d \tag{6.1}$$

- 在假定 gm 为恒定的前提下：

$$C_c = C_i - \frac{A_n}{g_m} \tag{6.2}$$

- 根据 Ficker 定理

$$A_n = \frac{g_s}{1.6}(C_a - C_i) \tag{6.3}$$

以及前文 4.1 提到的气孔导度模型

$$g_s = g_0 + g_1 \frac{A_n}{C_a} f(D) \tag{6.4}$$

整合公式 (6.1), (6.2), (6.3) 和 (6.4)，气体交换耦合模型有很多更大尺度上的应用，例如 Duursma and Medlyn (2012) 和 wang1998a，可用于预测 An，gs 和蒸腾速率对主要环境驱动因子的响应（除土壤水分），并包含了主要的叶片性状（g1，V_{cmax}, J_{max}, R_d 以及他们的温度依赖性）。

[1] 详细内容请参考原文

6.1 Photosyn 函数

Photosyn 为耦合的光合–气孔导度模型，基于 Farquhar 光合模型和 Ball-Berry 气孔导度模型。

```
Photosyn(VPD = 1.5, Ca = 400, PPFD = 1500,
         Tleaf = 25, Patm = 100,RH = NULL,
         gsmodel = c("BBOpti", "BBLeuning",
                     "BallBerry", "BBdefine"),
         g1 = 4, g0 = 0, gk = 0.5, vpdmin = 0.5,
         D0 = 5, GS = NULL, BBmult = NULL,
         alpha = 0.24, theta = 0.85, Jmax = 100,
         Vcmax = 50, gmeso = NULL, TPU = 1000,
         alphag = 0, Rd0 = 0.92, Q10 = 1.92,
         Rd = NULL, TrefR = 25, Rdayfrac = 1,
         EaV = 82620.87, EdVC = 0, delsC = 645.1013,
         EaJ = 39676.89, EdVJ = 2e+05, delsJ = 641.3615,
         GammaStar = NULL, Km = NULL, Ci = NULL,
         Tcorrect = TRUE,returnParsOnly = FALSE,
         whichA = c("Ah", "Amin", "Ac", "Aj"))

Aci(Ci, ...)
```

因为是光合气孔导度模型的耦合，故而参数与之前的函数相同，参见 3.5.2 和 4.4 部分的内容。

6.1.1 Photosyn 使用举例

```
library(plantecophys)
# 仅使用下面几个参数运行模型
#（其他参数使用默认值）
# 利用已测量或计算的参数
Photosyn(VPD=2, g1=4, Ca=500)
```

```
##         Ci    ALEAF        GS    ELEAF       Ac       Aj   Ap   Rd VPD Tleaf
## 1 369.3981 14.19466 0.1706377 3.412753 15.12654 17.03685 1000 0.92   2    25
##    Ca       Cc PPFD Patm
## 1 500 369.3981 1500  100
```

```
# 部分参数相同，而某一参数或某几个参数不同时，
# 可以将不同的参数设置为一个序列（vectors）
r <- Photosyn(VPD=seq(0.5, 4, length=25),
              Vcmax=50, Jmax=100)
with(r, plot(VPD, ALEAF, type='l'))
```

不同 VPD 下的光合速率见 6.1。

```
# 设定叶肉导度的拟合
run1 <- Photosyn(PPFD=seq(50,1000,length=25),
                 gmeso=0.15, Vcmax=40, Jmax=85)
with(run1, plot(PPFD, GS, type='l'))
```

```
# 运行 ACi 曲线数据（提供 Ci 值而不是计算）
arun1 <- Aci(Ci=seq(50, 1200, length=101),
             Vcmax=40, Jmax=85)
arun2 <- Aci(Ci=seq(50, 1200, length=101),
```

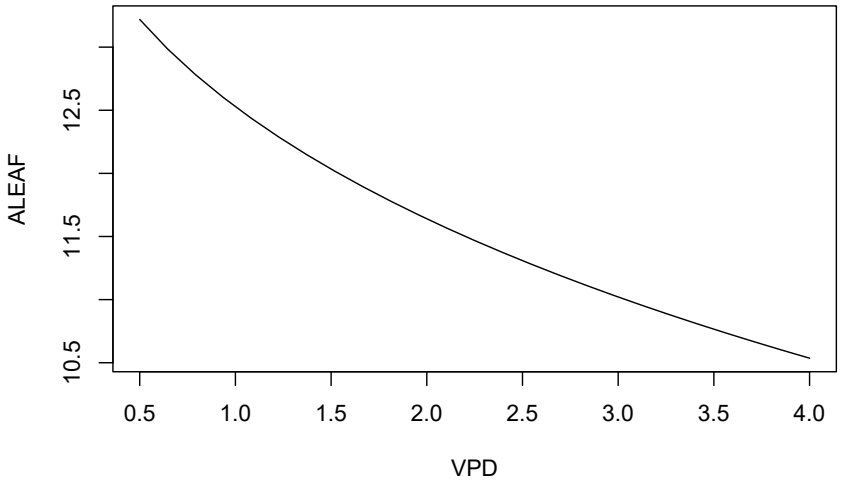


图 **6.1:** VPD VS. An

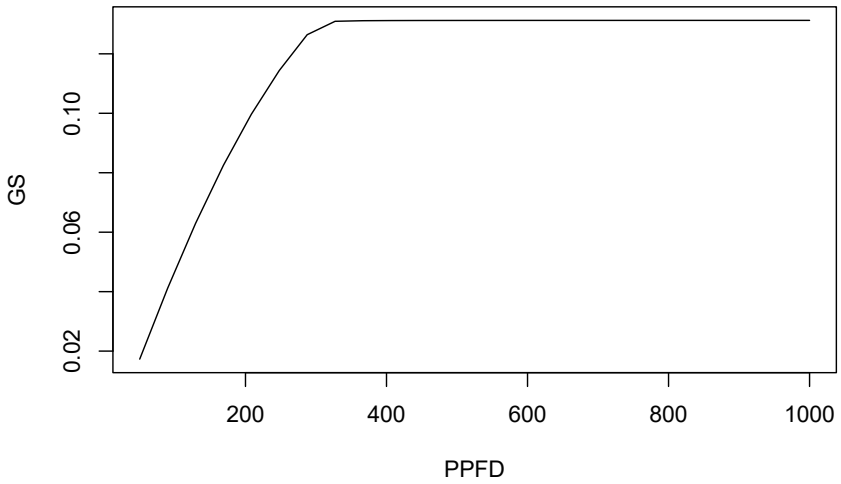


图 **6.2:** PPFD VS. GS

```
            Vcmax=30, Jmax=70)
with(arun1, plot(Ci, ALEAF, type='l'))
with(arun2, points(Ci, ALEAF, type='l', lty=5))
```

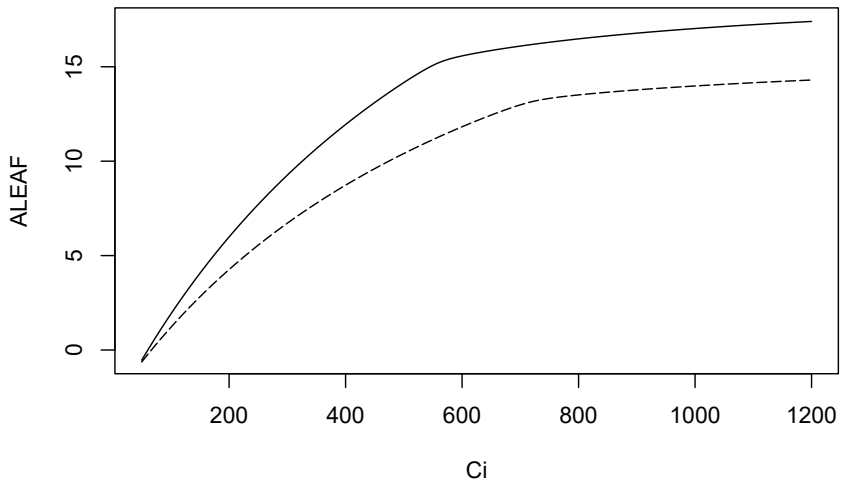


图 **6.3:** PPFD VS. GS

不同 ci 下的光合速率见 **??**。

```
# 找出 CO2 需求和供给的交叉点
# 设定部分参数
gs <- 0.2 # 对水的气孔导度
Ca <- 400 # 外部 CO2
gctogw <- 1.57 # 转换系数
gc <- gs / gctogw # 对 CO2 的气孔导度
# 需求曲线
p <- Aci(seq(60,500,length=101), Ca=400)
```

```
# 提供气孔导度及交叉点
g <- Photosyn(GS=gs, Ca=Ca)
# 交叉点可视化
par(yaxs="i")
with(p, plot(Ci, ALEAF, type='l',
             ylim=c(0,max(ALEAF))))
with(g, points(Ci, ALEAF, pch=19, col="red"))
abline(gc * Ca, -gc, lty=5)
legend("topleft", c(expression(
  "Demand:"~~A==f(C[i])),
   expression("Supply:"~~A==g[c]*(C[a]-C[i])),
              "Operating point"),
   lty=c(1,5,-1),pch=c(-1,-1,19),
   col=c("black","black","red"),
   bty='n', cex=0.9)
```

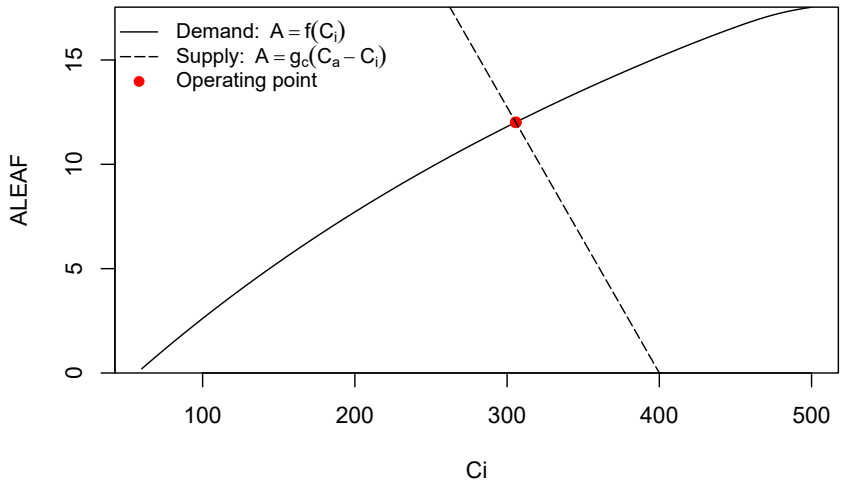


图 **6.4:** supply VS. demand

需求与供给的作图 6.4。

6.2 PhotosynEB 函数

使用同 `Photosyn`，只不过使用能量平衡法来计算叶温。目前版本尚存在 bug，不能直接提供 GS 来计算，否则会出现未知错误。

6.3 PhotosynTuzet 函数

同样为光合气孔导度耦合模型，只不过基于 Tuzet et al. (2003) 的气孔导度模型，现将其描述如下：

$$g_{co2} = g_0 + \frac{aA}{C_i - \Gamma} f_{\Psi v} \tag{6.5}$$

其中 g_{co2} 为对 CO_2 的气孔导度，g_0 是残余导度（residual conductance）(在补偿点时限制 g_{CO2} 的值），A 为光合速率，C_i 为胞间二氧化碳浓度，Γ 为 CO_2 补偿点，a 是经验系数。

一个根据经验的逻辑方程用于描述气孔对水势的敏感性：

$$f_{\Psi v} = \frac{1 + exp(s_f \Psi_f)}{1 + exp(s_f(\Psi_f - \Psi_v))} \tag{6.6}$$

Ψ_v 是指叶片水势，Ψ_f 是参比势能，该方程假设在水势接近于 0 时，气孔对叶片水势不敏感，并且气孔随着 Ψ_v 的下降快速关闭。Ψ_f 和 s_f (曲线形状参数) 依赖于不同物种的形态学适应以及生长过程中所处的环境。

6.3.1 PhotosynTuzet 的参数

除 Tuzet et al. (2003) 模型中使用的参数外，其他参数都继承了 Photosyn 的参数

```
PhotosynTuzet(g1 = 8, Ca = 400, psis = 0,
              kl = 2, sf = 3, psif = -2,
...)
```

- g1：斜率参数，要远比 fitBB 中的大。
- Ca：外界大气 CO_2 浓度
- psis，土壤水势（Mpa）。
- kl：Leaf-specific hydraulic conductance（叶片导水参数 $(mmol \cdot m^{-2} \cdot s^{-1} \cdot MPa^{-1})$，叶片蒸腾量、叶片面积及叶片水势计算）。
- sf：曲线形状参数。
- 在气孔导度为最大值 50% 时的叶片水势。

7

RHtoVPD 函数

该函数为一系列的工具，用于温度、湿度不同单位之间的换算 Jones (1993)：

```
# RH 转为 VPD
RHtoVPD(RH, TdegC, Pa = 101)

# VPD 转为 RH
VPDtoRH(VPD, TdegC, Pa = 101)

#  饱和水气压计算
esat(TdegC, Pa = 101)

#  露点温度转换为 VPD
DewtoVPD(Tdew, TdegC, Pa = 101)

#  基于叶温的 VPD 转换为基于空气温度的 VPD
VPDleafToAir(VPD, Tleaf, Tair, Pa = 101)

#  基于空气温度的 VPD 转换为基于叶温的 VPD
VPDairToLeaf(VPD, Tair, Tleaf, Pa = 101)

#  基于叶温的相对湿度转换为基于空气温度的相对湿度
RHleafToAir(RH, Tleaf, Tair, Pa = 101)
```

```
#  基于空气温度的相对湿度转换为基于叶温的相对湿度
RHairToLeaf(RH, Tair, Tleaf, Pa = 101)
```

注意事项及单位：

部分参数的转换需要准确的大气压用于计算，因此，有大气压选项的参数必须填入实际值。

RH：相对湿度 (%) 。

TdegC：温度 (℃) (叶片或温度)。

Pa：大气压 (kPa)

VPD：水气压亏缺 (kPa)。

Tdew：露点温度 (℃) 。

Tleaf：叶温 (℃) 。

Tair：空气温度 (℃) 。

8

光响应曲线的拟合

光响应曲线模型有很多，主要分为四大类，直角双曲线，非直角双曲线，指数以及直角双曲线修正模型，我们分别对这四类进行阐述。

8.1 直角双曲线模型

Baly (1935) 提出了直角双曲线模型，它的表达式为：

$$P_n = \frac{\alpha I \ P_{nmax}}{\alpha I + P_{nmax}} - R_d \tag{8.1}$$

- 其中，P_n 为净光合速率；
- I 为光强；
- α 为光响应曲线在光强为 0 时的斜率，即光响应曲线的初始斜率，也称之为初始量子效率；
- P_{nmax} 为最大净光合速率；
- R_d：为暗呼吸速率。

对 (8.1) 求导可知其导数大于 0，也就是直角双曲线是一个没有极值的渐近线，因此，无法由 (8.1) 求得最大光合速率的饱和光强[1]。

因此就需要使用弱光条件下 ($\leq 200\ \mu mol \cdot m^{-2} \cdot s^{-1}$) 的数据得到表观量子

[1] 直角双曲线和非直角双曲线模型类似，如果测量时饱和光强之后光合速率不下降，则计算的 P_{nmax} 则远大于实测值。

效率（apparent quantum efficiency，AQE），利用非线性最小二乘法估算 P_{nmax}，然后利用 ZiPiao (2010) 的式 (8.2) 求解 I_{sat}，

$$P_{nmax} = AQE \times I_{sat} - R_d \tag{8.2}$$

但此方法测得的光饱和点远小于实测值，我们采用 $0.7P_{nmax}$ Zhang et al. (2009)、$0.9P_{nmax}$ Huang et al. (2009)、或其他设定的值来的来估算 I_{sat}。

8.1.1 直角双曲线模型的实现

若没有安装 `minpack.lm`, 则需要首先：

```
install.packages("minpack.lm")
```

具体实现过程如下：

```
# 调用非线性拟合包 minpack.lm, 也可以直接使用 nls
library(minpack.lm)
# 读取数据，同 fitaci 数据格式
lrc <- read.csv("./data/lrc.csv")
lrc <- subset(lrc, Obs > 0)

# 光响应曲线没有太多参数，
# 直接调出相应的光强和光合速率
# 方便后面调用
lrc_Q <- lrc$PARi
lrc_A <- lrc$Photo

# 采用非线性拟合进行数据的拟合
lrcnls <- nlsLM(lrc_A ~ (alpha * lrc_Q * Am) *
                (1/(alpha * lrc_Q + Am)) - Rd,
```

```
              start=list(Am=(max(lrc_A)-min(lrc_A)),
              alpha=0.05,Rd=-min(lrc_A))
)
fitlrc_rec <- summary(lrcnls)

# 补偿点时净光合速率为 0,
# 据此利用 uniroot 求解方程的根
Ic <- function(Ic){(fitlrc_rec$coef[2,1] * Ic *
    fitlrc_rec$coef[1,1]) * (1/(fitlrc_rec$coef[2,1] *
    Ic + fitlrc_rec$coef[1,1])) - fitlrc_rec$coef[3,1]
}

uniroot(Ic, c(0,50))$root
```

```
## [1] 3.650053
```

```
# 根据饱和点定义, 0.75 最大光合速率为饱和点,
# 也可以是其他比例
# 据此利用 uniroot 求解方程的根

Isat <- function(Isat){(fitlrc_rec$coef[2,1] *
        Isat * fitlrc_rec$coef[1,1]) *
    (1/(fitlrc_rec$coef[2,1] * Isat +
          fitlrc_rec$coef[1,1])) - fitlrc_rec$coef[3,1] -
    0.75 * fitlrc_rec$coef[1,1]
}
# 求值区间根据具体实验确定

uniroot(Isat, c(0,2500))$root
```

```
## [1] 700.0946
```

```
# 使用 ggplot2 进行作图并拟合曲线
library(ggplot2)

light <- data.frame(lrc_Q = lrc$PARi, lrc_A = lrc$Photo)

p <- ggplot(light, aes(x = lrc_Q, y = lrc_A))

p1 <- p + geom_point(shape = 16, size = 3, color = "green") +
  geom_smooth(method="nls", formula = y ~ (alpha * x * Am) *
  (1/(alpha * x + Am)) - Rd, se = FALSE,
  method.args =
  list(start = c(Am=(max(lrc_A)-min(lrc_A)),
  alpha=0.05,Rd=-min(lrc_A)),
  aes(x =lrc_Q, y = lrc_A, color='blue', size = 1.2))
  ) +
  labs(y=expression(paste("photosynthetic rate  ",
       "(", mu, mol%.%m^-2%.%s^-1, ")")),
       x=expression(paste("PAR ",
       "(", mu, mol%.%m^-2%.%s^-1, ")"))
       )


# 自定义坐标轴
p1 + scale_x_continuous(breaks = seq(0, 2100, by = 200)) +
  scale_y_continuous(breaks= round(light$lrc_A)) +
  theme(axis.text.x  = element_text(
    size = 10, angle=30, vjust=0.5),
    axis.text.y = element_text(size = 10),
    axis.title.x = element_text(size = 12, face = 'bold'),
    axis.title.y = element_text(size = 12, face = 'bold')
  )
```

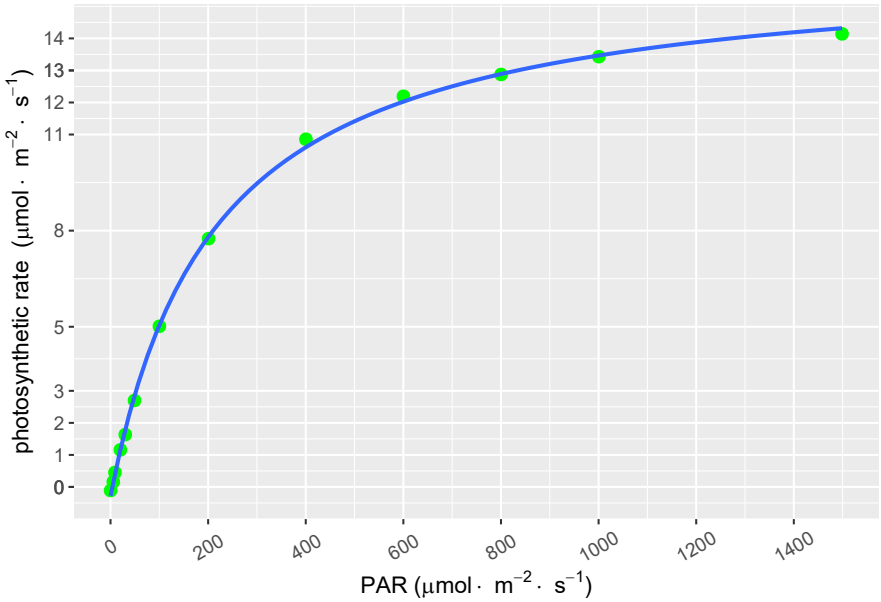


图 8.1: 直角双曲线模型拟合

代码目的见注释，其实现过程主要分三步：

- 数据的导入，这与之前相同，具体格式方法参考前文 **??**。
- 光响应曲线的拟合，使用到了非线性模型 nlsLM，也可以使用 nls，具体实现方法请查看参考文档。
- 求饱和点和补偿点，补偿点的计算根据其定义，净光合速率为 0，求解模型在一定区间的根来计算，而饱和点则较为麻烦，若使用式 (8.2) 计算，那么饱和点远远低于我们实际需求的，因此，我们使用了 $0.75\mathrm{P}_{nmax}$ 来计算，求得目标区间的根。当然也可以采用其他比例来作为饱和点光合速率。

最终的数据拟结果如图 8.1 所示，拟合的参数及结果见表 8.1。

表 8.1: 直角双曲线计算参数

	Estimate	Std. Error	t value	Pr(>|t|)
Am	16.6721752	0.1522849	109.480151	0.0000000
alpha	0.0783312	0.0026774	29.256870	0.0000000
Rd	0.2810926	0.0789338	3.561117	0.0051716

8.2 非直角双曲线模型

Thornley (1976) 提出了非直角双曲线模型，它的表达式为：

$$P_n = \frac{\alpha I + P_{nmax} - \sqrt{(\alpha I + P_{nmax})^2 - 4\theta\alpha I P_{nmax}}}{2\theta} - R_d \tag{8.3}$$

其中，θ 为表示曲线弯曲程度的曲角参数，取值为 $0 \leq \theta \leq 1$。其他参数意义同式 (8.1)。同样如同直角双曲线模型，式仍然没有极值，无法求得 I_{sat}，可以仍然参考直角双曲线模型的方式进行计算。

8.2.1 非直角双曲线模型的实现

```
library(minpack.lm)

# 读取数据，同 fitaci 数据格式
lrc <- read.csv("./data/lrc.csv")
lrc <- subset(lrc, Obs > 0)

# 光响应曲线没有太多参数，
# 直接调出相应的光强和光合速率
# 方便后面调用
lrc_Q <- lrc$PARi
lrc_A <- lrc$Photo

# 非直角双曲线模型的拟合
lrcnls <- nlsLM(lrc_A ~
                (1/(2*theta))*
                (alpha*lrc_Q+Am-sqrt((alpha*lrc_Q+Am)^2 -
                4*alpha*theta*Am*lrc_Q))- Rd,
```

```
                start=list(Am=(max(lrc_A)-min(lrc_A)),
                alpha=0.05,Rd=-min(lrc_A),theta=1))

fitlrc_nrec <- summary(lrcnls)

# 光补偿点
Ic <- function(Ic){
  (1/(2 * fitlrc_nrec$coef[4,1])) *
    (fitlrc_nrec$coef[2,1] * Ic + fitlrc_nrec$coef[1,1] -
    sqrt((fitlrc_nrec$coef[2,1] * Ic + fitlrc_nrec$coef[1,1]
    )^2 -  4 * fitlrc_nrec$coef[2,1] *
    fitlrc_nrec$coef[4,1] * fitlrc_nrec$coef[1,1] * Ic)) -
    fitlrc_nrec$coef[3,1]
}

uniroot(Ic, c(0,50))$root
```

```
## [1] 2.234292
```

```
# 光饱和点
Isat <- function(Isat){
  (1/(2 * fitlrc_nrec$coef[4,1])) * (fitlrc_nrec$coef[2,1] *
  Isat + fitlrc_nrec$coef[1,1] - sqrt(
  (fitlrc_nrec$coef[2,1] * Isat +fitlrc_nrec$coef[1,1])^2 -
  4*fitlrc_nrec$coef[2,1] * fitlrc_nrec$coef[4,1] *
  fitlrc_nrec$coef[1,1] * Isat)) -
  fitlrc_nrec$coef[3,1] - (0.9*fitlrc_nrec$coef[1,1])}

uniroot(Isat, c(0,2000))$root
```

```
## [1] 1596.286
```

```
# 使用 ggplot2 进行作图并拟合曲线
library(ggplot2)
light <- data.frame(lrc_Q = lrc$PARi, lrc_A = lrc$Photo)

p <- ggplot(light, aes(x = lrc_Q, y = lrc_A))

p1 <- p + geom_point(shape = 16, size = 3, color = "green") +
  geom_smooth(method="nls", formula = y ~
  (1/(2*theta))*(alpha*x+Am-sqrt((alpha*x+Am)^2 -
   4*alpha*theta*Am*x))- Rd, se = FALSE,
   method.args = list(start = c(Am=(max(lrc_A)-min(lrc_A)),
   alpha=0.05, Rd=-min(lrc_A), theta=1),
    aes(x =lrc_Q, y = lrc_A, color='blue', size = 1.2))
) +
  labs(y=expression(paste("photosynthetic rate  ",
          "(", mu, mol%.%m^-2%.%s^-1, ")")),
       x=expression(paste("PAR ",
           "(", mu, mol%.%m^-2%.%s^-1, ")")))

# 自定义坐标轴
p1 + scale_x_continuous(breaks = seq(0, 2100, by = 200)) +
  scale_y_continuous(breaks= round(light$lrc_A)) +
   theme(axis.text.x  = element_text(
    size = 10, angle=30, vjust=0.5),
    axis.text.y = element_text(size = 10),
    axis.title.x = element_text(size = 12, face = 'bold'),
    axis.title.y = element_text(size = 12, face = 'bold')
  )
```

最终的数据拟结果如图 8.2 所示，拟合的参数及结果见表 8.2。单纯从作图来看，本例数据使用非直角双曲线与散点图重合程度更高。

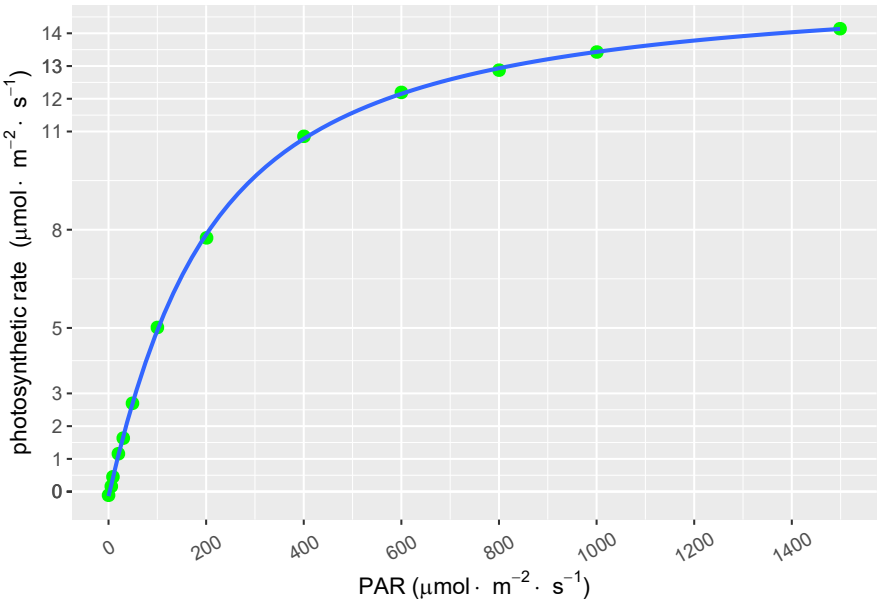


图 8.2: 非直角双曲线模型拟合

表 8.2: 非直角双曲线计算参数

	Estimate	Std. Error	t value	Pr(>|t|)
Am	15.8017296	0.1513064	104.435285	0.0000000
alpha	0.0658067	0.0020216	32.551422	0.0000000
Rd	0.1461717	0.0420800	3.473659	0.0070082
theta	0.3700908	0.0493403	7.500783	0.0000369

8.3 指数模型

光合指数模型较多，我们此处使用的指数函数的模型 Prado and Moraes (1997)，其表达式为：

$$P_n = P_{nmax}[1 - e^{-b(I-I_C)}] \tag{8.4}$$

其中，I_c 为光补偿点，e 为自然对数的底，b 为常数，其他参数意义同 (8.4)。同样，该方程仍然是没有极值的函数，但我们可以直接求得光补偿点。

8.3.1 指数模型的实现

```
library(minpack.lm)

# 读取数据，同 fitaci 数据格式
lrc <- read.csv("./data/lrc.csv")
lrc <- subset(lrc, Obs > 0)

# 光响应曲线没有太多参数，
# 直接调出相应的光强和光合速率
# 方便后面调用
lrc_Q <- lrc$PARi
lrc_A <- lrc$Photo

# 模型的拟合
lrcnls <- nlsLM(lrc_A ~ Am*(1-exp((-b)*(lrc_Q-Ic))),
                start=list(Am=(max(lrc_A)-min(lrc_A)),
                           Ic=5, b=1)
                )
fitlrc_exp <- summary(lrcnls)
```

```
# 光饱和点
Isat <- function(Isat){fitlrc_exp$coef[1,1]*
    (1-exp((-fitlrc_exp$coef[3,1])*(Isat-
    fitlrc_exp$coef[2,1])))-0.9*fitlrc_exp$coef[1,1]}

uniroot(Isat, c(0,2000))$root
```

```
## [1] 558.6038
```

```
## 拟合图形
library(ggplot2)
light <- data.frame(lrc_Q = lrc$PARi, lrc_A = lrc$Photo)

p <- ggplot(light, aes(x = lrc_Q, y = lrc_A))

p1 <- p +
  geom_point(shape = 16, size = 3, color = "green") +
  geom_smooth(method="nls", formula =
    y ~ Am*(1-exp((-b)*(x -Ic))),
    se = FALSE, method.args = list(
    start = c(Am=(max(lrc_A)-min(lrc_A)),
    Ic=5, b=0.002), aes(x =lrc_Q, y = lrc_A,
    color='blue', size = 1.2))
  ) +
  labs(y=expression(paste("photosynthetic rate  ",
          "(", mu, mol%.%m^-2%.%s^-1, ")")),
       x=expression(paste("PAR ",
           "(", mu, mol%.%m^-2%.%s^-1, ")")))
```

```
# 自定义坐标轴
p1 + scale_x_continuous(breaks = seq(0, 2100, by = 200)) +
  scale_y_continuous(breaks= round(light$lrc_A)) +
   theme(axis.text.x  = element_text(
    size = 10, angle=30, vjust=0.5),
    axis.text.y = element_text(size = 10),
    axis.title.x = element_text(size = 12, face = 'bold'),
    axis.title.y = element_text(size = 12, face = 'bold')
  )
```

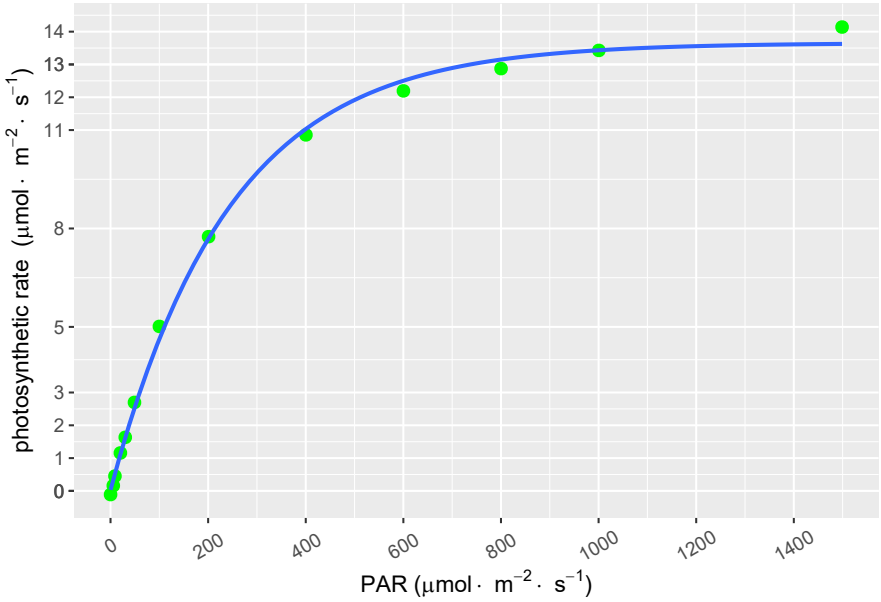


图 8.3: 指数模型拟合

最终的数据拟结果如图 8.3 所示，拟合的参数及结果见表 8.3。

表 8.3: 指数模型计算参数

	Estimate	Std. Error	t value	Pr(>|t|)
Am	13.6547568	0.1723363	79.233185	0.0000000
Ic	-0.5133438	2.3370250	-0.219657	0.8305573
b	0.0041183	0.0002012	20.467032	0.0000000

8.4 直角双曲线的修正模型

ZiPiao (2010) 直角双曲线修正模型的表达式如式 (8.5) 所示：

$$P_n = \alpha \frac{1-\beta I}{1+\gamma I} I - R_d \tag{8.5}$$

其中，β 和 γ 为系数，β 光抑制项，γ 光饱和项，单位为 $m^2 \cdot s \cdot \mu mol^{-1}$，其他参数与上文相同，因为该式 (8.5) 存在极值，因此，必然存在饱和光强和最大净光合速率，分别用式 (8.6) 和式 (8.7) 求得。

$$I_{sat} = \frac{\sqrt{\frac{(\beta+\gamma)}{\beta} - 1}}{\gamma} \tag{8.6}$$

$$P_{nmax} = \alpha \left(\frac{\sqrt{\beta+\gamma} - \sqrt{\beta}}{\gamma} \right)^2 - R_d \tag{8.7}$$

该模型的优点为拟合结果中光饱和点和最大净光合速率均接近实测值，还可以拟合饱和光强之后光合速率随光强下降段的曲线。

8.4.1 直角双曲线修正模型的实现

```
library(minpack.lm)

# 读取数据，同 fitaci 数据格式
lrc <- read.csv("./data/lrc.csv")
lrc <- subset(lrc, Obs > 0)

# 光响应曲线没有太多参数，
# 直接调出相应的光强和光合速率
# 方便后面调用
```

```
lrc_Q <- lrc$PARi
lrc_A <- lrc$Photo

# 模型的拟合
lrcnls <- nlsLM(lrc_A ~ alpha * ((1 -
              beta*lrc_Q)/(1 + gamma * lrc_Q)) * lrc_Q - Rd,
                start=list(alpha = 0.07, beta = 0.00005,
                           gamma=0.004, Rd = 0.2)
                )
fitlrc_mrec <- summary(lrcnls)

# 饱和点计算
Isat <-  sqrt((fitlrc_mrec$coef[2,1] + fitlrc_mrec$coef[3,1])/
              fitlrc_mrec$coef[2,1]-1)/fitlrc_mrec$coef[3,1]
# 补偿点计算
Ic <- (
  -(fitlrc_mrec$coef[3, 1] * fitlrc_mrec$coef[4, 1] -
  fitlrc_mrec$coef[1, 1]) - sqrt((fitlrc_mrec$coef[3, 1] *
  fitlrc_mrec$coef[4, 1] - fitlrc_mrec$coef[1, 1])^2-
  (4 * fitlrc_mrec$coef[1, 1] * fitlrc_mrec$coef[2, 1] *
  fitlrc_mrec$coef[4, 1])))/
  (2*fitlrc_mrec$coef[1,1]*fitlrc_mrec$coef[2,1])

## 拟合图形
library(ggplot2)
light <- data.frame(lrc_Q = lrc$PARi, lrc_A = lrc$Photo)

p <- ggplot(light, aes(x = lrc_Q, y = lrc_A))

p1 <- p +
  geom_point(shape = 16, size = 3, color = "green") +
  geom_smooth(method="nls", formula =
```

表 8.4: 直角双曲线修正模型计算参数

	Estimate	Std. Error	t value	Pr(>|t|)
alpha	0.0730858	0.0021209	34.460183	0.0000000
beta	0.0000501	0.0000133	3.776115	0.0043751
gamma	0.0040622	0.0001955	20.773916	0.0000000
Rd	0.2156186	0.0543505	3.967190	0.0032685

```
   y ~ alpha * ((1 -
              beta*x)/(1 + gamma * x)) * x - Rd,
    se = FALSE, method.args = list(
    start = c(alpha = 0.07, beta = 0.00005,
              gamma=0.004, Rd = 0.2),
    aes(x =lrc_Q, y = lrc_A,
    color='blue', size = 1.2))
  ) +
  labs(y=expression(paste("photosynthetic rate  ",
          "(", mu, mol%.%m^-2%.%s^-1, ")")),
       x=expression(paste("PAR ",
           "(", mu, mol%.%m^-2%.%s^-1, ")")))


# 自定义坐标轴
p1 + scale_x_continuous(breaks = seq(0, 2100, by = 200)) +
  scale_y_continuous(breaks= round(light$lrc_A)) +
   theme(axis.text.x  = element_text(
    size = 10, angle=30, vjust=0.5),
    axis.text.y = element_text(size = 10),
    axis.title.x = element_text(size = 12, face = 'bold'),
    axis.title.y = element_text(size = 12, face = 'bold')
  )
```

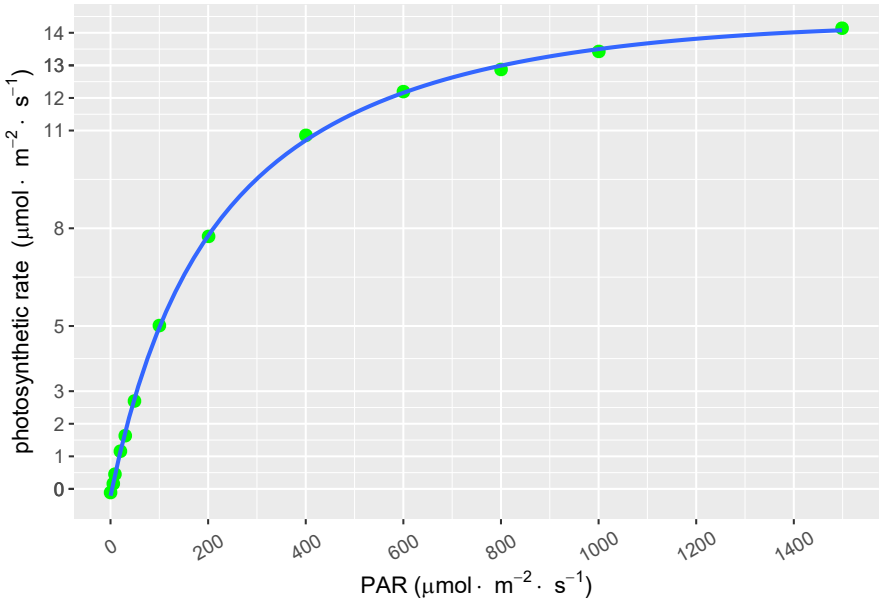


图 8.4: 直角双曲线修正模型拟合

尽管修正模型可以方便的计算饱和点和补偿点，但如同 Lobo et al. (2013) 所指出，双曲线模型对其结果的计算常有超出生态学意义范围的值[2]，因此对模型的选择不能一概而论，需根据实际情况而选择。

[2]例如本例的数据结果

9

关于非线性拟合的初始值

在解释初始值之前我们首先需要了解一个数学上的概念——迭代，

> “迭代法”也称“辗转法”是一种不断用变量的旧值递推新值的过程。

用通俗但不是特别严谨的说法可解释为：每次执行这种算法时，程序都会从原值（也就是我抄的上面迭代法定义的旧值）推出一个新值。

之所以先介绍这个迭代，原因很简单，非线性拟合就是通过迭代的方法，需要对每一个变量**最初的估计值进行不断的迭代，得到一个向一个点收缩或汇聚的值，这个估计值必须在实际值的一定范围内，程序通过不断调整这个值来改善拟合结果**。这就解释了上面的问题，初始值是让程序开始运行的前提，不然没法迭代，必须设定。我下面的内容将以 LI-6800 的光响应曲线的测试数据，使用非直角双曲线模型进行拟合来讲解具体的 R 中的一些实现方法，我们首先导入数据，然后再利用这些数据逐个举例不同的确定初始值的方式。

```
nls <- read.csv("data/nlstest.csv")


# 光响应曲线比较简单，我们将需要的数据直接提取，方便后面操作
```

```
lrc_Q <- nls$Qin
lrc_A <- nls$A
```

9.1 nlsLM 解决方案

nlsLM 来自于 Elzhov et al. (2016) 的 `minpack.lm`，利用 C 语言的 MINPACK 库，修改了 Levenberg-Marquardt 算法，在实际操作中，很多时候并不准确的输入初始值，他也能得出比较好的拟合结果。但结果未必完美，出现下面让人烦恼的报错：

> singular gradient matrix at initial parameter estimates

的概率会大大降低，而且尽管结果不如意，我们也可以利用他的结果缩小初始值的范围，继续尝试其他初始值。

例如下面的例子中，非直角双曲线的 Rd 的初始值我们可以利用暗呼吸的实测值大致估计，同理最大光合速率也是如此，剩下的分别为非直角双曲线曲率，我们暂定为 1，alpha 也暂定为 0.1，使用 `nlsLM` 进行拟合，结果如下：

```
library(minpack.lm)

lrcnls_lm <- nlsLM(lrc_A ~ (1/(2*theta))*
        (alpha*lrc_Q+Am-sqrt((alpha*lrc_Q+Am)^2 -
        4*alpha*theta*Am*lrc_Q))-
        Rd, start=list(Am=(max(lrc_A)-min(lrc_A)),
        alpha=0.1,Rd=-min(lrc_A),theta=0.8))
```

结果没有报错，看上去没有问题，那我们观察一下具体的拟合结果：

```
summary(lrcnls_lm)
```

```
##
## Formula: lrc_A ~ (1/(2 * theta)) * (alpha * lrc_Q + Am - sqrt((alpha *
##     lrc_Q + Am)^2 - 4 * alpha * theta * Am * lrc_Q)) - Rd
##
## Parameters:
##        Estimate Std. Error t value Pr(>|t|)
## Am    12.307570   0.406739  30.259 2.30e-10 ***
## alpha  0.045706   0.003423  13.352 3.09e-07 ***
## Rd     0.656638   0.132646   4.950 0.000791 ***
## theta  0.707522   0.079738   8.873 9.59e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1852 on 9 degrees of freedom
##
## Number of iterations to convergence: 8
## Achieved convergence tolerance: 1.49e-08
```

结果看上去还可以[1]。

9.2 作图比对法

模型很多参数可以用已有数据去估计，我们可以只来分析难以判断的参数，流程如下：

[1]有些时候结果并不理想，该方法并不是万无一失

- Rd、Am 等我们可以利用测量值来确定一个范围。
- 剩余的参数，我们也可以根据经验或文献来有一个大致的判断。
- 然后我们根据数学的方式来判断哪个参数对曲线形状影响最大（例如在分母上的参数，或者是乘以该参数，该参数可以显著改变计算结果，例如整体乘以或除以 0.1 还是 0.01，像 Rd 之类的参数本身就很小，多数公式中都是减去该值，对结果影响很小，我们通常直接使用实测值）。
- 将该参数取一系列值带入模型来求解净光合速率。
- 将计算的 A 值与光强进行作图，看我们计算的曲线与测量数据点的重合程度，必要时在修改其他参数，使曲线和散点重合度最好，重合程度最高的参数值即为我们需要的初始值。

9.2.1 实现过程

```
# 我们选择的模型，将其写为一个函数，用于计算净光合速率
expfct <- function(x, Am, alpha, Rd, theta) {
  (1/(2 * theta)) * (alpha * x + Am -
  sqrt((alpha * x + Am)^2 - 4 * alpha * theta * Am * x)) - Rd
}

# 我们的数据
test <- data.frame(x = lrc_Q, y = lrc_A)
```

```
# 先做实测数据的散点图
plot(y ~ x, data = test)

# 利用上面的函数，假定 alpha 的值为 0.8，看计算值与测量值重合程度
curve(expfct(x, Am = (max(lrc_A)-min(lrc_A)),
     alpha=0.8, Rd=-min(lrc_A), theta=0.8), add = TRUE
           )
```

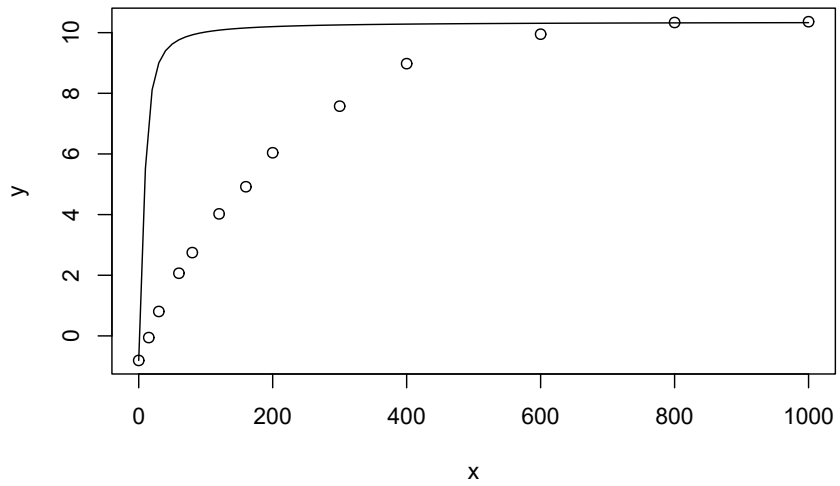


图 9.1: 初步判断 alpha 的初始值

观察上图 9.1 的结果可以看到，曲线在 0-600 的范围内，拟合值明显偏大，观察模型的方程式，以及其他起始值的设定方式，我们初步判断 alpha 的值偏大，于是乎我们将其改小观察，观察曲线和测量点的重合仍然不是很好，我们尝试修改 theta 值与 alpha 值（也即曲线高于测量点，则需要减小纵坐标的值，低于测量点，则需要增加该值，该过程省略，我大概设置了五分钟完成），最终得出的结果如下：

```
plot(y ~ x, data = test)
curve(expfct(x, Am = (max(lrc_A)-min(lrc_A)),
     alpha=0.06, Rd=-min(lrc_A), theta=0.82), add = TRUE)
```

图 9.2 尽管看上去效果仍然不满意，但我们可试着进行拟合，看能否得到显著差异的结果：

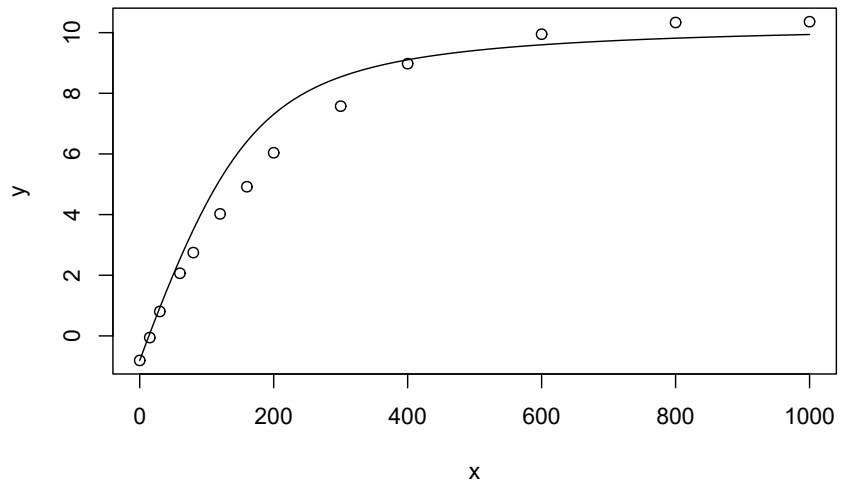


图 9.2: 初修正后断 alpha 的初始值

```
lrcnls_manual <- nls(lrc_A ~
        (1/(2*theta))*
        (alpha*lrc_Q+Am-sqrt((alpha*lrc_Q+Am)^2 -
                             4*alpha*theta*Am*lrc_Q))-
        Rd, start=list(Am=(max(lrc_A)-min(lrc_A)),
                       alpha=0.03,Rd=-min(lrc_A),theta=0.6))
summary(lrcnls_manual)
```

```
##
## Formula: lrc_A ~ (1/(2 * theta)) * (alpha * lrc_Q + Am - sqrt((alpha *
##     lrc_Q + Am)^2 - 4 * alpha * theta * Am * lrc_Q)) - Rd
##
## Parameters:
##        Estimate Std. Error t value Pr(>|t|)
## Am    12.307585   0.406741  30.259 2.30e-10 ***
```

```
## alpha  0.045706   0.003423  13.352 3.09e-07 ***
## Rd     0.656642   0.132646   4.950 0.000791 ***
## theta  0.707518   0.079739   8.873 9.59e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1852 on 9 degrees of freedom
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 4.601e-06
```

```
# 对拟合之后的结果作图，观察使用我们的估计值，
# 迭代的最终值与元数据的重合程度
plot(y ~ x, data = test, ylim = c(-2, 14))
curve(expfct(x, Am = 12.307586,
     alpha=0.045706, Rd= 0.656643, theta=0.707518), add = TRUE)
```

从 9.3 的呈现以及 F 检验的 p 值来讲，图形已经比较完美了。**也就是说尽管我们作图的时候看到重合度并不高，但是非线性拟合本来就是一个迭代的过程，只要我们的数据与真实值相差不大，还是能够得到完美结果的。**

9.2.2 直观展示

上面的表述太啰嗦，直接用下面的图形说明一下，其中 alhpa 的取值在此处选择从 0.01 到 0.07，每次增加 0.05，其他值分别为 Am = 12.31, Rd= 0.66, theta=0.71 （此处为展示效果和方便，将这些值直接按照拟合结果设定了，实际差别不大）

```
library(ggplot2)
library(purrr)
```

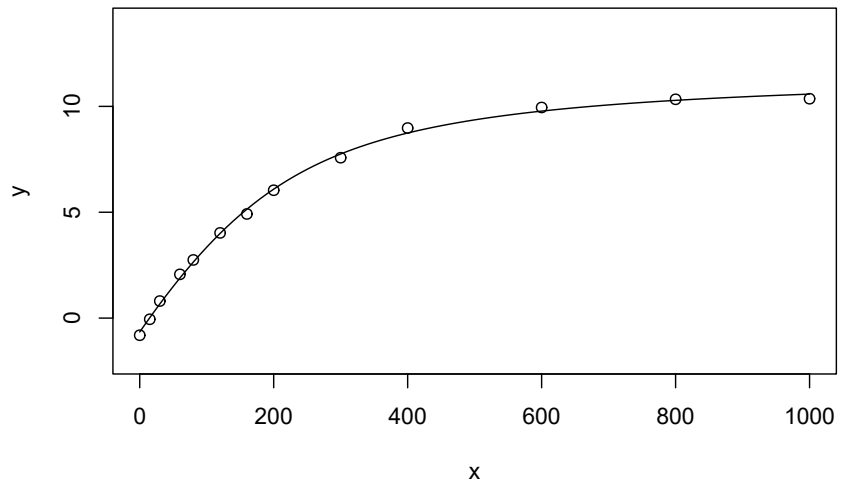


图 9.3: 检验作图法的初始值判断

```
lrc <- read.csv("data/nlstest.csv")

# 光响应曲线比较简单，我们将需要的数据直接提取，方便后面操作
lrc_Q <- lrc$Qin
lrc_A <- lrc$A
n <- length(lrc_A)


alp <- paste0("a=", seq(0.01, 0.07, by = 0.005))
alpn <- rep(alp, each = n)

expfct <- function(x, Am, alpha, Rd, theta) {(1/(2 * theta)) * (alpha * x + Am - sqrt
}

paras <- data.frame(alpha = rep(seq(0.01, 0.07, by = 0.005), each = n),
```

```
            x = rep(lrc_Q, n), Am = rep(12.31, n), Rd = rep(0.66, n),
            theta = rep(0.71, n))
y = unlist(pmap(paras, expfct))

show <- data.frame(x = rep(lrc_Q, 14),
            y = c(lrc_A, y),
            a = factor(c(rep("measured", n), alpn),
            level = c("measured", alp)
              ))

ggplot(data = show, aes(x, y, group = a, color=a)) +
  geom_point() +
  geom_smooth(se = FALSE)
```

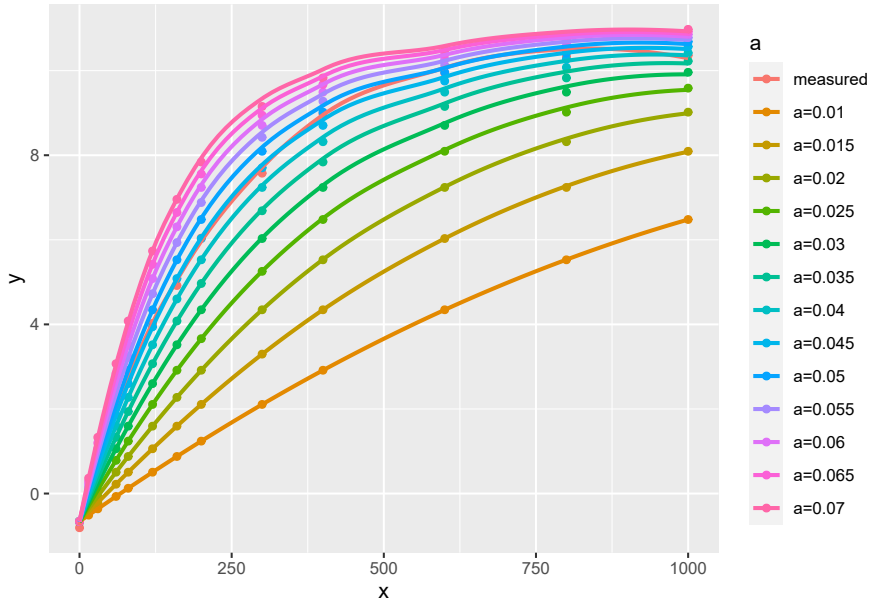


图 9.4: 多个 alpha 取值的差异

从上图 9.4 我们我们可以看到，实测值在 alpha =0.04 和 alpha = 0.05 两条

曲线之间，在 0.045 时最接近测量点，也就是我们把初始值设为 0.04 和 0.05 之间最接近，本例中可认为是 0.045，实际这三个值均可。

9.3 自动多次尝试法

该方法实际为使用 nls2 来实现，具体方法参考 Bouvier and Huet (1994) 的文章，可简单概括为使用一系列的起始值梯度（例如下面的代码中，alpha 的取值在 0.01 到 0.08 之间），然后软件循序使用不同的起始值，即排列组合所有的起始值序列，最终找到合适的值，具体实现如下：

```
library(nls2)
```

```
## Loading required package: proto
```

```
grid.test <- expand.grid(list(
  Am=c(12),
  alpha = seq(0.01, 0.08, by =0.01),
  Rd = seq(0, 3),
  theta=seq(0.1, 1, by = 0.1)
  ))

lrcnls2 <- nls2(lrc_A ~
        (1/(2*theta))*
        (alpha*lrc_Q+Am-sqrt((alpha*lrc_Q+Am)^2 -
                               4*alpha*theta*Am*lrc_Q))-
        Rd, start = grid.test, algorithm = "brute-force")
```

```
## Error in numericDeriv(form[[3L]], names(ind), env) :
```

```
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
## Error in numericDeriv(form[[3L]], names(ind), env) :
##   Missing value or an infinity produced when evaluating the model
```

```
summary(lrcnls2)
```

```
##
## Formula: lrc_A ~ (1/(2 * theta)) * (alpha * lrc_Q + Am - sqrt((alpha *
##     lrc_Q + Am)^2 - 4 * alpha * theta * Am * lrc_Q)) - Rd
##
## Parameters:
##        Estimate Std. Error t value Pr(>|t|)
## Am    12.000000   0.623023  19.261 1.27e-08 ***
## alpha  0.050000   0.006414   7.795 2.72e-05 ***
## Rd     1.000000   0.260153   3.844  0.00394 **
## theta  0.800000   0.102143   7.832 2.62e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3663 on 9 degrees of freedom
##
## Number of iterations to convergence: 320
## Achieved convergence tolerance: NA
```

通过结果可以看到，虽然和之前采用手动方法判定的结果比较接近，但是还是略有差异，可以看一下他们各自的结果同测量值的重合程度：

```
plot(lrc_Q, lrc_A)
lines(lrc_Q,  predict(lrcnls2), col="red")
lines(lrc_Q,  predict(lrcnls_manual), col="blue")
```

图 9.5 可以看到，使用 `nls2` 的拟合结果似乎和测量值更匹配，当然这只是第一印象，后续的判断还要进一步通过 F 检验、AIC、BIC 等统计方式才能判定。

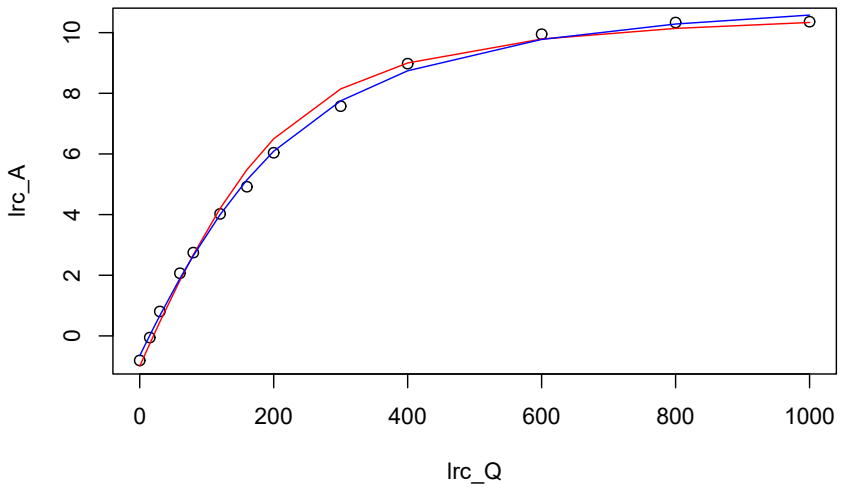


图 9.5: 两种方法结果的对比展示

9.4 小结

采用如上三种方式都可以有效的解决起始值的问题，`nlsLM` 操作上更易实现，对初始值的大小不敏感，但设置不能太离谱，否则仍然会报错。作图比对法操作上更麻烦一些，但是这种方式一定能得出合理的初始值设置。采用 `nls2` 类似于将手动作图方式自动化，类似于 SPSS 中非线性拟合中需要给出一个初始值的范围，且该范围不能过大。如有一定的经验，操作起来将非常迅速。

需要注意的是，这三种方法结合起来使用会更好，例如，即使使用 `nlsLM` 的结果不合理，也可以参考他们参数的范围（部分结果也可能是差异显著），然后将这些结果用于手动作图判定参数或者 `nls2` 中判定参数范围，或者使用作图法确定大致的范围，将该范围输入到 `nls2` 中，这样会节省时间，也更加方便。

10

LI-6800 的数据分析

10.1 数据格式

~~LI-6800 的数据前处理同 LI-6400 相同，如表10.1 所示的样式。~~

并非此处格式无效，只是有更简洁的数据读取方式，见 2.4.2.2。

表 10.1: 推荐 LI-6800 整理后数据样式

obs	time	elapsed	date	TIME	E
1	1471425750	0.0	20160817 17:22:30	1471425726	0.0007614
2	1471425855	104.5	20160817 17:24:14	1471425829	0.0009221
3	1471426153	403.1	20160817 17:29:13	1471426144	0.0009900
4	1471426244	494.0	20160817 17:30:44	1471426235	0.0010249
5	1471426335	585.1	20160817 17:32:15	1471426326	0.0010632
6	1471426426	676.0	20160817 17:33:46	1471426417	0.0011190

10.2 LI-6800 与 LI-6400 使用时的差别

`plantecophys` 使用时建立在 LI-6400XT 基础之上的软件包，因此在 LI-6800 代码中，需要改动的是 fitaci、fitacis 及 fitBB 中的 varnames 选项，也就是将 LI-6400XT 的表头改为 LI-6800 的表头。以 fitaci 函数为例：

```
fitaci(aci, varnames =
       list(ALEAF = "A", Tleaf = "Tleaf", Ci = "Ci",
         PPFD  = "Qin", Rd = "Rd"))
```

10.3 光响应曲线注意事项

光响应曲线的拟合相对简单，仅需要光强和光合速率的值，其中需要修改的部分仅为光强的赋值部分，在文件名一致的前提下，修改如下代码即可：

```
lrc_Q <- lrc$Qin
lrc_A <- lrc$A
```

10.4 LI-6800 RACiR™ 的测量与拟合

在评估作物性状时，V_{cmax} 及 J_{max} 时非常有用，传统的 A–Ci 曲线测量要求植物叶片要在一定浓度 CO_2 下适应几分钟后完成测量，这样的测量有几个缺点：

- 测量时间长，一条曲线至少需要 20 – 30 min，样本量多，重复多时，这种方法几乎没有可行性。
- 整个测量过程中，时间长，酶的激活状态会有变化，叶绿体会移动，气孔的开度也会发生变化。

而 LI-6800 独有的 auto control 功能在算法上允许用户自定义 CO_2 的起始浓度和种植浓度、变化方式（线性或其他）、所花费的时间，再加上其 IRGAs 极快的响应频率，使得短时间内的 A–Ci 的测量成为现实，即快速 CO_2 响应曲线 RACiR™ 测量实验，该功能使得 5 min 内测量 A–Ci 曲线成为可能。该方法的实现可参考 Stinziano et al. (2017) 的文章。

Stinziano et al. (2018) 针对 RACiR™ 技术的疑问做了解答并提出了准确测量的建议，概括如下：

- 首先，采用 100 ppm/min 的变化速率是与标准方法重合度最高的测量。
- 其次，明确研究问题，目前已有研究表明 Vcmax 与 Jmax 的计算结果与标准测量方法结果无显著差异。
- 任何条件的改变，都需要做空叶室校准，例如：流速，气体浓度变化方向、温度，斜率等。
- 空叶室校准与叶片测量采用严格的同一次校准，因为 IRGA 的漂移，需要再次匹配时，或者环境条件改变时，需要重新做空叶室校准。是否需要匹配，可通过不加叶片的最初状态查看，此时 A 值应接近为 0，reference 和 sample 气体浓度读数接近相等。
- IRGA 分析器使用 5 此多项式进行校准，推荐使用 1 次到 5 次多项式进行拟合，然后根据 BIC 指数来确定最合适的空叶室校准系数（即非参数拟合的模型选择的问题）。确定最合适的浓度变化范围。通常需要去掉最初和最后 30 s 的数据。

- 最小化校准和测量值之间的水分摩尔分数差异。甚至有可能需要控制 reference 或 sample 的水的摩尔分数而不是 Vpdleaf。通过预实验来确定最合适的 CO_2 变化范围和随时间的斜率。

10.5 racir 软件包实现 RACiR™ 数据分析

可能软件包作者没有更改叶面积的需求，他的软件包只支持原始数据的处理，但这对有更改需求的客户来讲没那么友好，而且这样导致了大量代码的重复，我根据作者原来的代码，结合我自己数据分析的习惯，重新制作了一份软件包，下面代码实现是基于我的软件包实现的，如有需求，也可使用作者原来的代码实现

```
devtools::install_github("zhujiedong/racir")
library(racir)
```

软件包的函数很多，但可以分为下面几类：

- 首先使用 `find_cut` 来查看使用数据的范围（排除野点）。
- 使用 `racircheck` 来检查校正数据 (非必须，但建议进行数据质量检查)
- 使用 `racircal` 进行数据的校准

10.5.1 实现方法 1

手上暂时没有原始数据格式的 racir 数据，因而现在使用以前的 csv 数据进行操作（空叶室测量数据和带叶片测量数据转为 csv 格式）。

首先检查空叶室校准：

```
library(racir)
# only check the empty chamber data here
em1 <- read.csv("./data/racirem/em-1.csv")
le1 <- read.csv("./data/racirle/le-1.csv")
find_cut(em1)
```

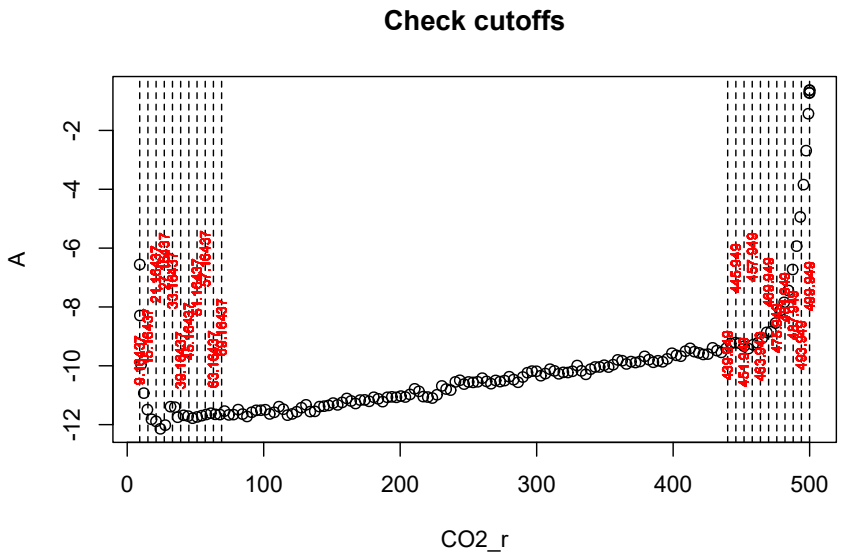


图 10.1: 找出最合理的校准曲线数据范围

上述代码查看参比室 CO_2 浓度在两者之间的数据，确定后面代码使用的 `mincut`, `maxcut` 范围。此处我选择的范围如下问代码：

```
racircheck(em1, mincut = 21 , maxcut = 463 )
```

没有问题可直接进行校准

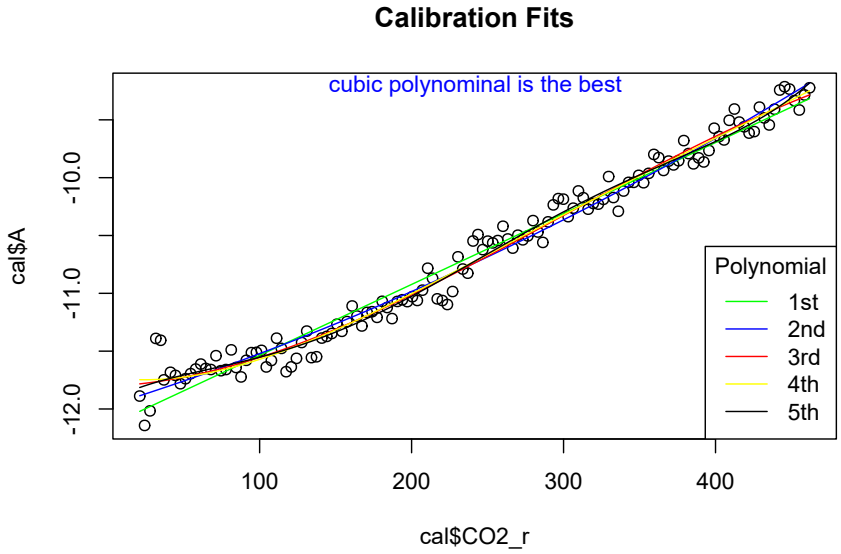
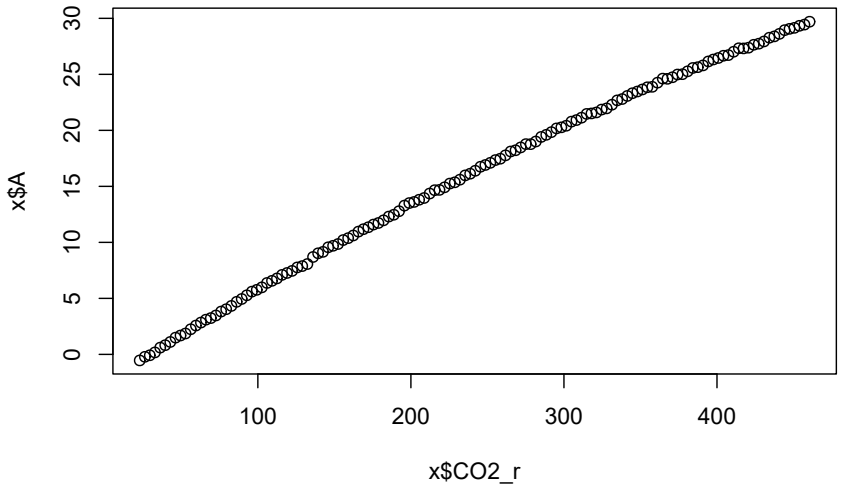


图 10.2: 校准曲线查看

```
x <- racircal(em1, le1, mincut = 21, maxcut = 463)
plot(x$CO2_r, x$A)
```

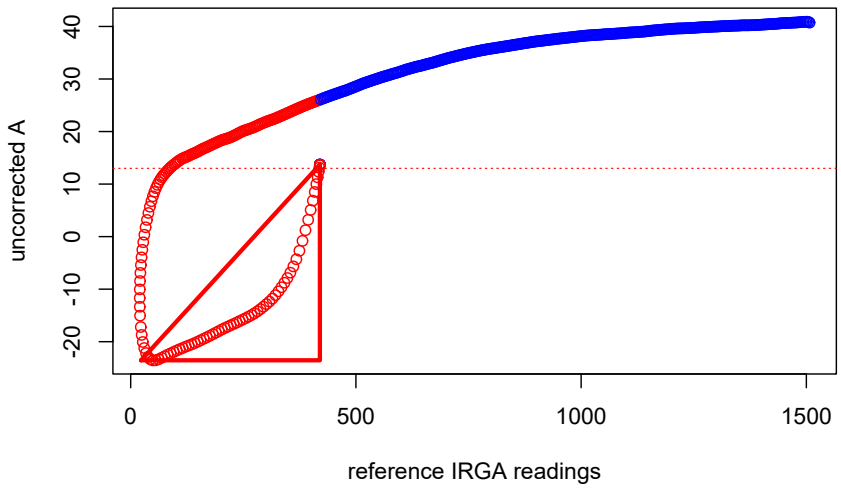


前两个参数分别为空叶室数据，带叶片测量数据，后面的最大最小值为 `racircheck` 确定，剩余工作就是使用 `plantecophys` 进行分析，在此不再重复这个过程。

注意：RACiR 的本意是表型研究中求 Vcmax 和 Jmax，因而，请忽略 Rd 等参数的计算结果

10.5.2 实现方法 2

以上的实现方法是基于肉眼判断，删除不需要的数据，自动化程度较低，同时不适合 Coursolle et al. (2019) 这样的方法:



直接批量删除是不可取的。容易出错或者残留无效数据，针对这个情况我进行的修改和测试为：使用 BP 进行测量，目前我已经完成 BP 相关的程序，本节的示例则是基于我实测的验证数据来进行（相关测试已经录好视频，可能后面公司公众号会推，如有 BP 自动测量 RACiR 的需要，请跟我们公司联系）。这个程序叶室基于 Coursolle et al. (2019) 的方法，测量步骤和修改如下：

1. 在接近外界的 CO_2 浓度下进行诱导，控制 reference 的 CO_2 420 ppm ，控制 H_20 为 20 $mmol \cdot mol^{-1}$，**此处不做修改。**
2. 使用自动控制，在 2 min 时间内，将 CO_2 的 reference 气路浓度，下降到 20，然后等待 10-15 s 的时间。**此处的修改为，不记录这些数据，减少数据处理的麻烦。**例如 @ref(fig: racir-steady-start) 中三角形区域的数据。
3. 运行自动控制，将 CO_2 的 reference 浓度由 20 上升到 1520 ppm（仅作为测试范围，并非绝对），**本步骤不做修改，记录数据，此为所需要数据。**

实际上主要的修改就是将这个步骤自动化，需要人为操作的步骤仅仅是点 start 按键启动整个测量。

对于软件包的修改：

主要的修改：

- 独立的 `xls_read` 函数用于读取 excel 格式数据并修改叶面积。
- 独立的 “清洗” 空叶室数据的函数 `tidy_data_empty`。
- 独立的 “清洗” 带叶片测量数据的函数 `tidy_data`。
- 将数据的清洗使用 A 值的一个上限和一个下限来解决。

使用的方法如下（有使用标准 ACi 测量数据的比较内容）：

10.5.2.1 查看相邻数据的差值设置

数据的清洗其实比较简单，主要利用相邻的两个测量 A 值的差值来进行，为方便确定差值，针对空叶室和带叶片测量的数据，单独设定了两个函数，来确定合理的过滤范围，这样说太抽象，我们导入数据后直接看图：

1. 导入数据：

此函数本质上是 `xlconnect_read`，如无需修改叶面积，则使用默认值。如果需要修改叶面积，需要输入 S 参数，叶面积修改值必须和 A 的值一一对应，否则因为叶面积错误，则导致其他的错误，例如上面修改叶面积为原来的一半：

```
library(racir)
library(plantecophys)

empty <- xls_read("data/racir-empty.xlsx")
leaf <- xls_read("data/racir-leaf.xlsx")

#  标准的 aci 数据
aci <- xls_read("data/aci-curve.xlsx")

half_leaf <- xls_read("data/racir-leaf.xlsx", S = leaf$S/2)
head(half_leaf$A/leaf$A)
```

```
## [1] 2 2 2 2 2 2
```

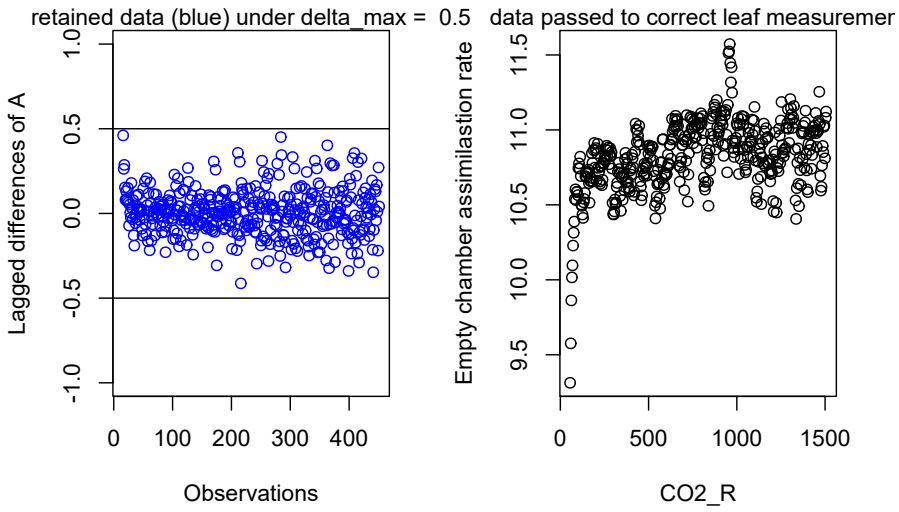
```
tail(half_leaf$A/leaf$A)
```

```
## [1] 2 2 2 2 2 2
```

修改面积仅作演示用，如无需修改，则不要输入面积选项。

2. 关于空叶室的数据过滤范围的选取:

```
check_delta_empty(empty, delta_max = 0.5, lower_A = -1)
```

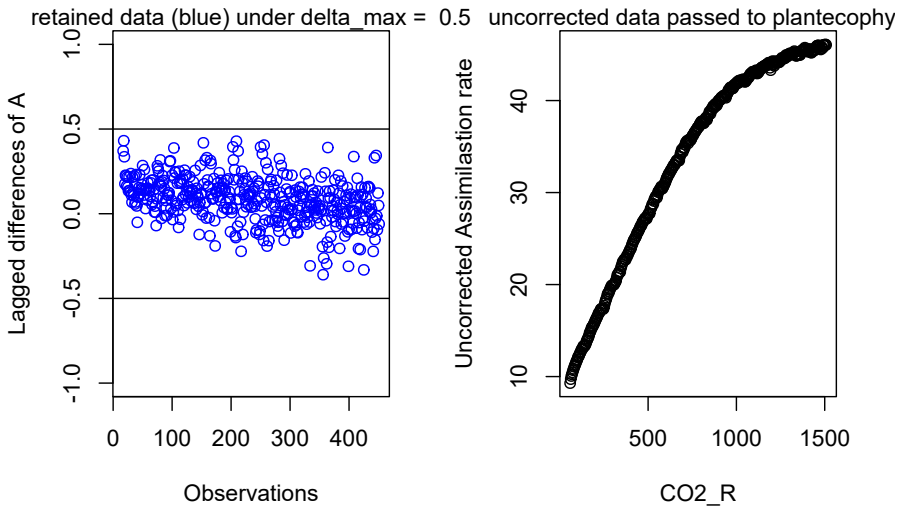


这里首先不要被第二幅图的波动所迷惑，主要是坐标轴差异太小。然后第一幅图中，这里设定了一个最大最小光合速率的范围（阈值，upper_A 和 lower_A），默认最大值是 2 以及最小值是 -2，上面我使用了最小值为 -1，也就是删除小于 -1 的光合速率，最大值 2 并不是删除，二是寻找 $-2 < A < 2$ 范围的数据，从这里面找一个最小值，保留这一行的数据到测量数据结束。

另一个阈值删除相邻两个 A 值过大差值的数据的阈值（delta_max），对于这个阈值，LI-6800 其实并不敏感，因为本身数据非常稳定。

3. 关于带叶片测量的数据过滤范围的选取:

```
check_delta(leaf, delta_max = 0.5, lower_A = -1)
```



带叶片的数据同上，没有特殊的参数，从右图的数据来看，他在修正之前的光合速率，符合我们的预期。

4. 确定范围后数据的清洗:

```
tidy_leaf <- tidy_data(leaf, delta_max = 0.5, lower_A = -1)
tidy_empty <- tidy_data_empty(empty, delta_max = 0.5, lower_A = -1)

# 修正带叶片测量数据
z <- correct_leaf(tidy_empty, tidy_leaf)
```

4. 拟合后的数据：

```
fit_racir <- fitaci(z,
                varnames =
                  list(
                    ALEAF = "A",
                    Tleaf = "Tleaf",
                    Ci = "Ci",
                    PPFD  = "Qin",
                    Rd = "Rd"
                  ))
fit_normal <- fitaci(aci,
                varnames =
                  list(
                    ALEAF = "A",
                    Tleaf = "Tleaf",
                    Ci = "Ci",
                    PPFD  = "Qin",
                    Rd = "Rd"
                  ))
```

结果的比较：

```
knitr::kable(data.frame(fit_racir$pars))
```

	Estimate	Std..Error
Vcmax	78.245344	0.2479143
Jmax	170.501593	0.3781222
Rd	2.252787	0.0670844

```
knitr::kable(data.frame(fit_normal$pars))
```

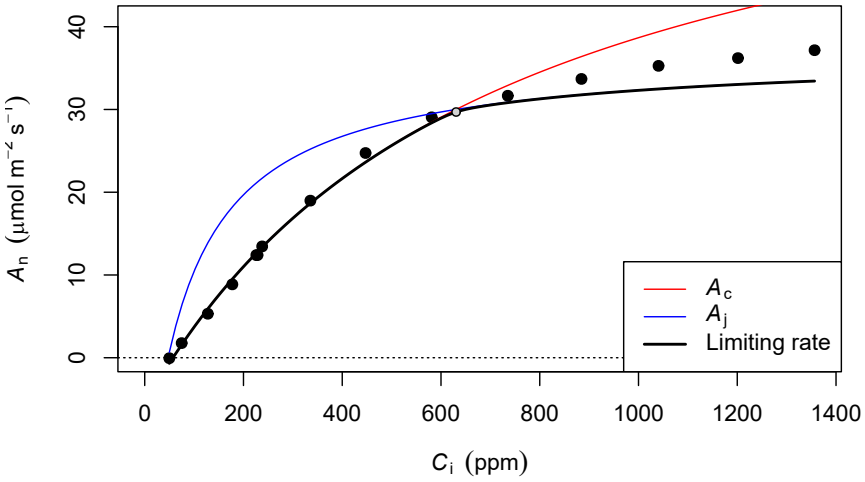


图 10.3: ACi 拟合结果的图形

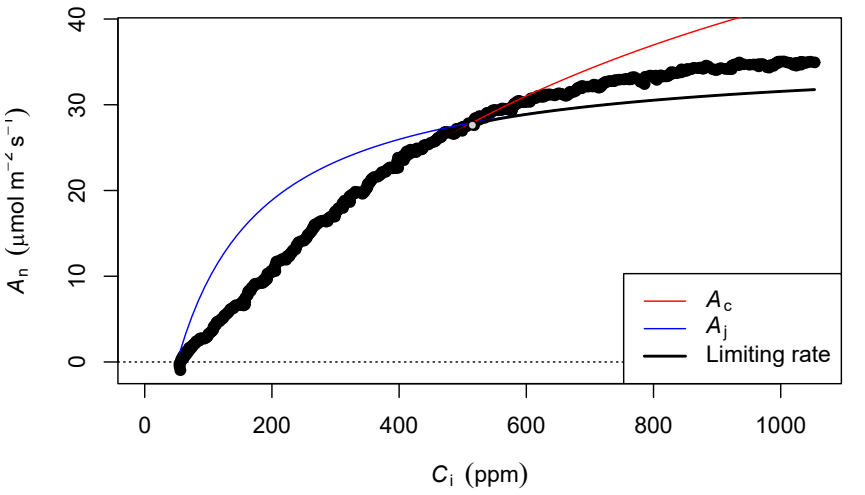


图 10.4: RACiR 拟合结果的图形

	Estimate	Std..Error
Vcmax	71.547963	1.5236176
Jmax	169.701477	2.4151240
Rd	1.348372	0.3872127

此处的数据和视频里的数据以及函数的参数略有差别，因为我修改了一下 racir 软件包的过滤条件，使得筛选数据更符合要求。本身因为不是同一时间测量，测量 RACiR 时叶片的气孔导度略到，因此，其 V_{cmax} 略大符合预期。

数据的重合程度（实际上 RACiR 数据点要浓密的多，我使用了透明度为 80%，这样看上去仍然颜色非常深，这是 RACiR 采集数据点达到几百个的原因。ACi 测量点为 50-2000 ppm，RACiR 为 20-1520 ppm）：

```
library(RColorBrewer)
library(latex2exp)

line_col <- brewer.pal(2,"Set1")

racir_col <- scales::alpha(line_col[1], 0.2)
aci_col <- line_col[2]

plot(
  z$Ci,
  z$A,
  cex = 2.3,
  col = racir_col,
  pch = 20,
  xlim = c(50, 1400),
  ylim = c(-1, 38),
  xlab = TeX('$C_i$ $(\\mu mol \\cdot mol^{-1})$'),
  ylab = TeX('A $(\\mu mol \\cdot mol^{-1})$')
)
points(aci$Ci,
```

```
        aci$A,
        cex = 2.3,
        col = aci_col,
        pch = 20)

legend(
  10,
  35,
  legend = c("RACiR","ACi"),
  col = c(racir_col, aci_col),
  pch = 20,
  bty = "n",
  pt.cex = 2.3
)
```

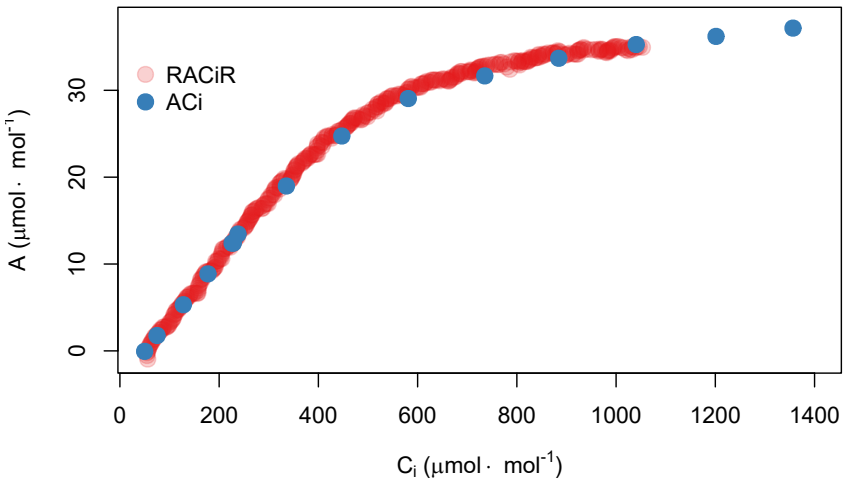


图 10.5: (#fig:acivsracir_bp)RACiR 与 ACi 数据点的重合程度（BP 测量）

10.5.3 数据的批量处理

10.5.3.1 readphoto 导入

如果数据的导入是使用 readphoto 批量处理，那么所在数据文件有 files 列，作为不同文件数据的区分，那么可以使用向量化处理方式实现数据的分析，参考如下代码：

```
library(readphoto)
# read all empty data set
all_empty <- read_bat_6800("./empty_data")
# read all leaf data set
all_leaf <- read_bat_6800("./leaf_data")
listem  <-  split(all_empty, files)
listle <-  split(all_leaf, files)
cordata <- purrr::map2(listem, listle, racircal, mincut = 21, maxcut = 463)
```

所有的数据均保存在 cordata 中，特别注意 **此处要求 listem 与 listle 长度相同，因此，如果一个空叶室对多个带叶片测量数据，请将空叶室测量文件复制几份（注意对应顺序），使其每个带叶片测量的数据文件和空叶室测量数据文件数量相同，并且一一对应，例如 emtpy-1-1 数据文件对应 leaf-1-1**，文件命名前即制定好命名规则，方便后续数据处理。

10.5.3.2 处理大量 csv 文件

```
# suppose leaf and empty data in 2 files in disk D:
le=list.files("d:/racirle/")
em=list.files("d:/racirem/")

# construct the path of these csv files
pathle = paste0("d:/racirle/", le)
```

```
pathem = paste0("d:/racirem/", em)

# import all leaf data and empty data separately
xx <- lapply(pathle, read.csv)
yy <- lapply(pathem, read.csv)

# correct all the data
all_correc <- purrr::map2(yy, xx, racircal, mincut = 21, maxcut=463)

#  export all data to csv files, these file have the same name with leaf data
dirc <- dir.create("d:/data/")
finame <- "d:/data/"
finame <- paste0(finame, le)
purrr::walk2(all_correc,  finame, write.csv)
```

以上代码仅供参考，主要是实现批量导入并校正，在批量将校正后数据导出为 csv 文件，为方便，将导出的文件名字命名为同原来带叶片测量数据相同的文件名，。

10.6 批量计算 V_{cmax} 和 J_max

```
#use csv data above
csvdata <- lapply(all_correc, plantecophys::fitaci,
    varnames = list(ALEAF = "A", Tleaf = "Tleaf",
    Ci = "Ci", PPFD= "Q", Rd = "Rd"), fitmethod ="bilinear")

# use readphoto data above
photodata <- lapply(cordata, plantecophys::fitaci,
```

```
    varnames = list(ALEAF = "A", Tleaf = "Tleaf",
    Ci = "Ci", PPFD= "Q", Rd = "Rd"), fitmethod ="bilinear")
```

批量分析同样使用向量化来进行，避免了循环，效率比较高，也可以将拟合结果 `coef` 中的 V_{cmax} 和 J_max 结果批量导入文件，参考 @??imbycsv) 部分的代码。

10.7 LI-6800 RACiR™ 簇状叶的测量与拟合

Coursolle et al. (2019) 测量了簇状叶黑云杉和香脂冷杉两种簇状叶植物的 RACiR，其中的试验方法和结论值得在测量时借鉴，测量方法上：

- 簇状叶室体积远大于荧光叶室和其他叶室，使用的 CO_2 的变化为：15 min 内从 20 ppm 到 1520 ppm 的变化，即变化的速率为 $100 ppm \cdot min^{-1}$。但也测试了 200 - 800 ppm 的部分曲线。
- 拟合使用了测量的 Rd，测量方法为：控制 reference 气路在 420 ppm 的 CO_2 和 22 $mmol \cdot mol^{-1}$ 的 H_2O 浓度，控制温度为 25 C，诱导后测量 Rd。

得到了一些有帮助的结论：

- 使用更大的叶室测量 RACiR 是可行的（36 cm^2），叶室环境的控制需要通过预实验来确定。
- 该实验使用的 ACi 曲线测量时间在 30 到 36 min，而 RACiR 使用的**完整的二氧化碳的浓度范围**时，曲线耗时最大的时间接近 22 min。但使用 200 - 800 ppm 范围的变化，则时间可以下降 50%，**这些部分范围的测量则可以应用于植物胁迫和表型平台的研究**。
- 实验结果证明只要 match 的调整值保持不变即无需进行空叶室校准（也就是无需匹配的意思，实际的时间间隔取决于仪器的状态），但最新的 range match 功能可有效的增加空叶室校准的时间间隔（新功能，作者试验时尚未推出该功能）。
- 作者建议最好测量暗呼吸的速率，以获得最佳的 Vcmax 和 Jmax 计算结果。如果有第二台光合仪来测量则可有效的缩短测量时间。

10.7.1 数据的拟合

10.7.1.1 数据文件的命名

因为作者读取数据和修正数据都依赖于正则表达式，因此，所有文件的命名规则必须满足以下的要求：

- 空叶室校准的文件必须以 “mpty” 起始
- RACiR 的文件必须以 “fast” 起始
- 常规 ACi 测量必须以 “slow” 起始
- Rd 测量数据文件必须以 “dark” 起始

例如作者示例数据分别命名为：

Empty_2.xlsx，Fast.xlsx，Slow.xlsx，Dark.xksx，实际使用的正则表达式为:

```
pattern_empty       = "^(mpty)+.xls",
pattern_rapidACi    = "^(fast)+.xls",
pattern_standardACi = "^(slow)+.xls",
pattern_dark        = "^(dark)+.xls",
```

如不满足要求，自然会出现错乱。准备好命名的数据文件之后，那么可以把他们放在一个文件夹内。

10.7.1.2 匹配的问题

若进行了匹配，要重新开始空叶室校准，RapidACi 软件包采用的规则是，使用 Match 分组下的 time 来区分，首先使用该时间和空叶室一致的 RACiR 数据配对校准，若无一致的数据，则选择该时间和 RACiR 测量相差最短的空叶室数据来校准 RACiR 数据。

10.7.2 拟合过程

作者给出了详细的代码和数据，此处不再重复和演示，请参考：

github 演示[1]

发表文章的附加材料[2]

注：今天测试了他附加材料的代码，有可能遇到问题，或者有一些注意事项，请参考下面的代码（本地已经有数据了，因此并未执行下载的代码，改为本地读取）。

```
# 安装及加载大量的依赖
remotes::install_github("ManuelLamothe/RapidACi")
if (!require(tidyverse))
  install.packages("tidyverse")
if (!require(readxl))
  install.packages("readxl")
if (!require(XLConnect))
  install.packages("XLConnect")
if (!require(plantecophys))
  install.packages("plantecophys")

library(RapidACi)
library(XLConnect)
library(tidyverse)
library(readxl)
library(plantecophys)
```

[1] https://github.com/ManuelLamothe/RapidACi

[2] https://www.frontiersin.org/articles/10.3389/fpls.2019.01276/full#supplementary-material

```
#build_list:
# 测量最好直接存放 “data” 文件夹下
# 然后建立了 4 个匹配规则，分别是:
#pattern_empty = "(mpty).*\\.xls",
#pattern_rapidACi = "(fast).*\\.xls",
#pattern_standardACi = "(slow).*\\.xls",
#pattern_dark = "(dark).*\\.xls",

# 如果后面需要修改叶面积，那么需要提供额外的 dataframe，其中必须有两列:
# sample_ID, 以及其中的一列必须命名为 LeafArea_mm2

# 默认 match 的 time 列为 BN，若有其他用户自定义选项，请修改默认参数
# 空叶室校准的选择原则为：默认选择距离 racir 测量时间最近的空叶室（empty 开头的）测量
# 进行校准


list_files <- build_list()
list_files$sample_ID[1] <- "fastdemo"

# 了解一下 build_list 的作用
list_files

results <- Rapid_aci_correction(list_files)
results

# 诊断校准的结果
# 如果报错: unable to start png() device
# 那么可能是权限的原因无法建立 figure 文件夹
dir.create("./figure/", showWarnings = FALSE)
# delta 为 A diff 的差值，默认设定了 0.05
diagnose_sample(results, sample_name = "fastdemo")
```

```
# 添加 racir
Raci <- results[["fastdemo"]]$Raci

# range sheet measurements 绝对引用 I17
Dark <-
  read_excel(
    "./ds/Dark.xlsx",
    range = "Measurements!I17",
    col_types = "numeric",
    col_names = FALSE
  )

Raci$Rd <- pull(Dark)

fRaci <- fitaci(Raci, useRd=TRUE)
par(mgp=c(2.5,1,0), family="serif", ps=14)
plot(fRaci, las=1,xlim=c(0,1200), ylim=c(0,30))
```

10.8 RACiR™ 分析的手动实现

以下内容是我之前写的内容，部分代码的实现和上述软件包相似，仅供参考，尤其是当您使用 racir 软件包报错时，可以参考下文代码手动实现 RACiR 数据的分析。

```
# 加载需要的软件包 ------------------------------------------------------------

library("plantecophys")

# 准备数据 --------------------------------------------------------------------

#-----------------------------------------------------
# 也可以使用上文中的 xls_read() 来直接读取 excel 数据
#-----------------------------------------------------
empty_uc <- read.csv("./data/racir-csv/uncorr_emp500.csv")
leaf_uc <- read.csv("./data/racir-csv/uncorr_leaf500.csv")
aci <- read.csv("./data/racir-csv/aci_ex.csv")

# 防止读入空白行
empty_uc <- subset(empty_uc, obs > 0)
leaf_uc <- subset(leaf_uc , obs > 0)
aci <- subset(aci, obs > 0)

# 空叶室校准的系数计算 --------------------------------------------------------

# 观察空叶室未校准数据 reference 对 A 的图形
plot(empty_uc$CO2_r, empty_uc$A)
# 选取线性部分用于校准
locator()
```

```
# 执行 locator 命令后，在上图中的目标位置选点，
# 选好后按 esc 可以返回所选点的坐标（选点即为在
# 预期位置鼠标单击）
# 根据上面的点，利用二氧化碳的值过滤掉不需要的数据
# 只要在线性范围内选点，拟合结果相差很小

empty_ct <- empty_uc[which(empty_uc$CO2_r >
  45.28 & empty_uc$CO2_r < 459.12), ]
plot(empty_ct$CO2_r, empty_ct$A)

# 采用 1~5 次多项式分别拟合
cal1st <- lm(A ~ CO2_r, data = empty_ct)
cal2nd <- lm(A ~ poly(CO2_r, 2), data = empty_ct)
cal3rd <- lm(A ~ poly(CO2_r, 3), data = empty_ct)
cal4th <- lm(A ~ poly(CO2_r, 4), data = empty_ct)
cal5th <- lm(A ~ poly(CO2_r, 5), data = empty_ct)
# 利用 BIC 找出最合理的校准方程
bics <- BIC(cal1st, cal2nd, cal3rd, cal4th, cal5th)
# noquote 也就是没引号，成为名字
best <- noquote(rownames(bics)[bics$BIC == min(bics$BIC)])
best


# 校准带叶片测量的数据 ------------------------------------------------------------

leaf_uc$A <- leaf_uc$A - predict(cal4th, leaf_uc)
leaf_uc$Ci <- ((leaf_uc$gtc - leaf_uc$E / 2) * leaf_uc$CO2_s -
                 leaf_uc$A) / (leaf_uc$gtc + leaf_uc$E / 2)

# 对校准后的数据进行作图，查看校准效果
plot(leaf_uc$CO2_r,
```

```
     leaf_uc$A,
     pch = 2,
     ylim = c(-20, 40))

# 选取带叶片测量的有效数据的范围
locator()

leaf_ct <- leaf_uc[which(leaf_uc$CO2_r > 13.6 &
   leaf_uc$CO2_r < 471), ]
plot(leaf_uc$CO2_r, leaf_uc$A)


# 拟合 ----------------------------------------------------------------------
racir <- fitaci(
  leaf_ct,
  varnames =
    list(
      ALEAF = "A",
      Tleaf = "Tleaf",
      Ci = "Ci",
      PPFD  = "Qin",
      Rd = "Rd"
    ),
  Patm = 84.09
)

slow <- fitaci(
  aci,
  varnames =
    list(
      ALEAF = "A",
      Tleaf = "Tleaf",
```

```
      Ci = "Ci",
      PPFD  = "Qin",
      Rd = "Rd"
    ),
  Patm = 84.09
)

# 查看拟合数据
racir$pars
slow$pars

# 对快速曲线作图拟合结果进行查看
plot(racir, linecols = c("green", "blue", "red"))
```

注意，注意数据表头的大小写，此处代码中，为处理数据的方便，我更改了大小写，分析自己的数据时需要注意。

10.9 多个速率的 RACiR 曲线研究

自 RACiR 技术诞生以来，极大的缩短了 Vcmax 以及 Jmax 的测量时间 (Stinziano et al. (2017))，但也引起了一系列争议，作者也对业内的质疑进行了一一的解答 (Stinziano et al. (2018))，但除了因为时间长短导致酶活性，叶绿体位置等差异外，RACiR 还能说明哪些问题呢？Joseph R. et al. (2019) 最新的研究给出一系列结论：

- 扩散限制（CO_2 总导度）和光呼吸导致了表观上的标准 ACi 曲线和 RACiR 测量之间的偏差，表明他们的差异是由**生物因子引起，而非仪器导致的人为误差**。
- 上述原因导致的二者之间的偏差，如果不进行修正，那么将显著的低估 Γ^*，除非使用多个速率的 RACiR 来修正。
- 较高速率的 RACiR 曲线会增大其与标准曲线之间的偏差，但这个差距在无光呼吸的条件下会减小。
- 因为光呼吸和气体扩散限制与物种相关，结合以上结论，可以使用多个速率的 RACiR 来估算对 CO_2 的总导度以及相对量的光呼吸速率。

一些可能的方向：

- 扩散限制影响 Cc 速率的变化，说明对具有较高总阻力与 CO_2 比值的物种，例如针叶物种，C4 植物，较高的阻力导致 RACiR 与标准 ACi 测量斜率更大的差异，或者测量的前提假设被破坏。
- RACiR 可检测到代谢中 CO_2 的滞后性，各种滞后性的检测对标准 ACi 测量也具有指示性。

文中利用 R 实现了光呼吸之后模型和气体扩散限制模型，本文内容主要对文献中附录材料的源码进行解释：

10.9.1 光呼吸滞后模型

为测试光呼吸的滞后性，作者使用一系列预先设定的参数，模拟了一条 ACc 曲线，假定 Rubisco 激活状态为 100%，并且在整个测量过程中气孔导度是不变的。然后使用这些参数来模拟 RACiRs 曲线，并且假定光呼吸分别需要 0, 15, 30, 60, 120 或 300 s 来对变化的 CO_2 进行响应，在实际效果上，这意味着 Cc 在最初的 0, 15, 30, 60, 120 或 300 内是不变的，最后我们对 Γ^* 和 Ci 使用线性回归进行计算。

10.9.1.1 构造基础数据

模型第一步，则是对需要使用的参数，根据文献和实际情况进行赋值，具体内容参考代码注释。

```
library(ggplot2)
library(plyr)
```

```
##
## Attaching package: 'plyr'

## The following object is masked from 'package:purrr':
##
##     compact
```

```
library(gridExtra)

# 对图例使用自定义颜色
gg_color_hue <- function(n) {
  hues = seq(15, 375, length = n + 1)
  hcl(h = hues, l = 65, c = 100)[1:n]
}
```

```
#Maximum Rubisco carboxylation rate in umol m-2 s-1
Vcmax <- 110

#Maximum Rubisco oxygenation rate in umol m-2 s-1;
#Ratio from Bernacchi et al. 2001. PCE 24:253-259
Vomax <- 0.29 * Vcmax
#Dark respiration in umol m-2 s-1
R <- 2
#Michaelis-Menten constant for Rubisco carboxylation in umol mol-1
Kc <- 404.9
#Michaelis-Menten constant for Rubisco oxygenation in mmol mol-1
Ko <- 278.4
#oxygen concentration in mmol mol-1
O2 <- 210
Kco <- Kc * (1 + O2 / Ko)
#Boundary layer conductance in mol m-2 s-1
BLC <- 2
#stomatal conductance in mol m-2 s-1
gsw <- 0.4
#mesophyll conductance in mol m-2 s-1
gm <- 1
#Chloroplastic CO2 in umol mol-1
Cc <- as.numeric(c(25:400))
#oxygenation rate in umol m-2 s-1
vo <- Vomax * O2 / (O2 + Ko * (1 + Cc / Kc))
#carboxylation rate in umol m-2 s-1
vc <- Vcmax * (Cc) / (Cc + Kco)
#Net CO2 assimilation in umol m-2 s-1
A <- vc - 0.5 * vo - R
#Apparent CO2 assimilation rate in umol m-2 s-1
Aapparent <- vc - 0.5 * vo
```

```
#Intercellular CO2 in umol mol-1
Ci <- A / gm + Cc
#Boundary layer CO2 in umol mol-1
Cb <- A / gsw + Ci
#Reference CO2 in umol mol-1
Cr <- A / BLC + Cb

# 根据 Cc 浓度的个数，构造向量
Counter <- as.numeric(c(1:length(Cc)))
# 也就是以秒计算的 cr 与时间的模型
RateCrmodel <- lm(Cr ~ Counter)
# 转化为分钟的 cr 的斜率
RateCr <- coef(RateCrmodel)[2] * 60

# 转换为分钟的边界层导度斜率
RateCbmodel <- lm(Cb ~ Counter)
RateCb <- coef(RateCbmodel)[2] * 60

# 转换为分钟的 ci 的斜率
RateCimodel <- lm(Ci ~ Counter)
RateCi <- coef(RateCimodel)[2] * 60

# 转换为分钟的 Cc 的斜率
RateCcmodel <- lm(Cc ~ Counter)
RateCc <- coef(RateCcmodel)[2] * 60
```

10.9.2 光呼吸滞后性代码

下面代码的目的是为得到 ACi 响应曲线受光呼吸延迟的影响，尤其是在临近补偿点时。

延迟模块

```
# 假定有 15s 延迟时的数据，即相比上面构造的 Cc 数据减少 15 个点
Cc15 <- as.numeric(c((min(Cc) + 15):max(Cc), rep(max(Cc), 15)))
vo15 <- Vomax * O2 / (O2 + Ko * (1 + Cc15 / Kc))
A15 <- vc - 0.5 * vo15 - R
Aapparent15 <- vc - 0.5 * vo15

#30 s 延迟数据
Cc30 <- as.numeric(c((min(Cc) + 30):max(Cc), rep(max(Cc), 30)))
vo30 <- Vomax * O2 / (O2 + Ko * (1 + Cc30 / Kc))
A30 <- vc - 0.5 * vo30 - R
Aapparent30 <- vc - 0.5 * vo30

#60s 延迟数据
Cc60 <- as.numeric(c((min(Cc) + 60):max(Cc), rep(max(Cc), 60)))
vo60 <- Vomax * O2 / (O2 + Ko * (1 + Cc60 / Kc))
A60 <- vc - 0.5 * vo60 - R
Aapparent60 <- vc - 0.5 * vo60

#120s 延迟数据
Cc120 <- as.numeric(c((min(Cc) + 120):max(Cc), rep(max(Cc), 120)))
vo120 <- Vomax * O2 / (O2 + Ko * (1 + Cc120 / Kc))
A120 <- vc - 0.5 * vo120 - R
Aapparent120 <- vc - 0.5 * vo120

#300s 延迟数据
Cc300 <- as.numeric(c((min(Cc) + 300):max(Cc), rep(max(Cc), 300)))
vo300 <- Vomax * O2 / (O2 + Ko * (1 + Cc300 / Kc))
A300 <- vc - 0.5 * vo120 - R
Aapparent300 <- vc - 0.5 * vo300
```

10.9.3 数据的构造

下面的代码主要是将上文最终计算的数据构造数据集，并导出。

```
Anet <- c(A, A15, A30, A60, A120, A300)
Aapp <-
  c(Aapparent,
    Aapparent15,
    Aapparent30,
    Aapparent60,
    Aapparent120,
    Aapparent300)
Ccfull <- rep(Cc, 6)
Cifull <- rep(Ci, 6)
Delay <-
  c(
    rep("0", length(A)),
    rep("15", length(A15)),
    rep("30", length(A30)),
    rep("60", length(A60)),
    rep("120", length(A120)),
    rep("300", length(A300))
  )
PRdata <- as.data.frame(cbind(Anet, Aapp, Ccfull, Cifull, Delay))
write.csv(PRdata, "./data/PRdata.csv")
```

10.9.4 光呼吸滞后性作图

下面的代码是将光呼吸的数据进行作图。

```
data <- read.csv("./data/PRdata.csv")
data$Ccfull <- as.numeric(data$Ccfull)
data$Delay <- as.factor(data$Delay)

# 净光合速率与 Cc 作图
AnetCc <- ggplot(data, aes(x = Ccfull, y = Anet, colour = Delay)) +
  geom_point() +
  labs(x = expression(C[c] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
  labs(colour = 'Delay (s)') +
  scale_x_continuous(limits = c(25, 100),
                     breaks = c(25, 40, 55, 70, 85, 100)) +
  scale_y_continuous(limits = c(-5, 5)) +
  scale_colour_brewer(palette = 'Spectral') +
  # 补偿点的参考线
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AnetCc
```

```
# 净光合速率与 Ci 作图
AnetCi <- ggplot(data, aes(x = Cifull, y = Anet, colour = Delay)) +
  geom_point() +
  labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
```

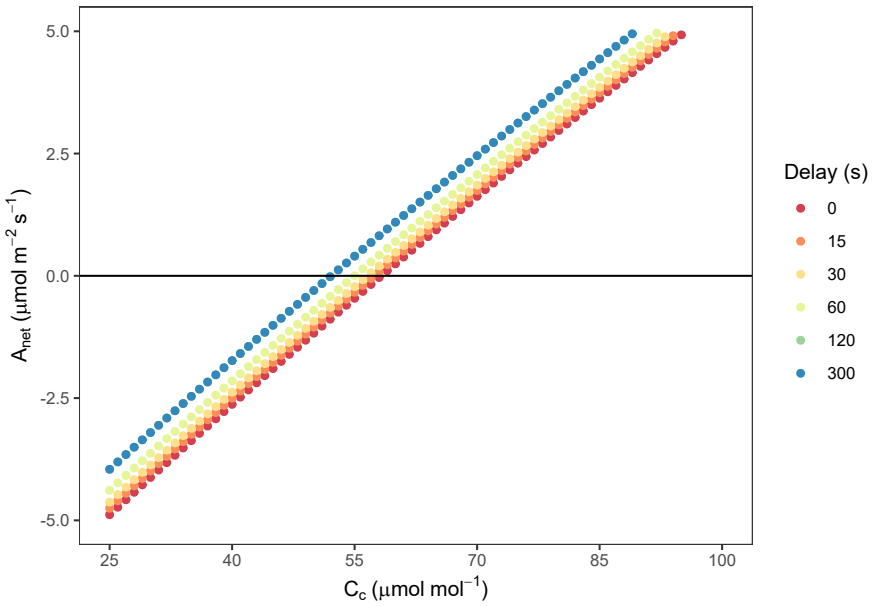


图 **10.6:** Anet VS. Cc

```
  -1
} * ")"),
y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
  -2
} ~ s ^ {
  -1
} * ")")) +
labs(colour = 'Delay (s)') +
scale_x_continuous(limits = c(25, 100),
                   breaks = c(25, 40, 55, 70, 85, 100)) +
scale_y_continuous(limits = c(-5, 5)) +
scale_colour_brewer(palette = 'Spectral') +
geom_hline(yintercept = 0) +
theme_bw() +
theme(panel.grid.major = element_blank(),
```

```
        panel.grid.minor = element_blank())
AnetCi
```

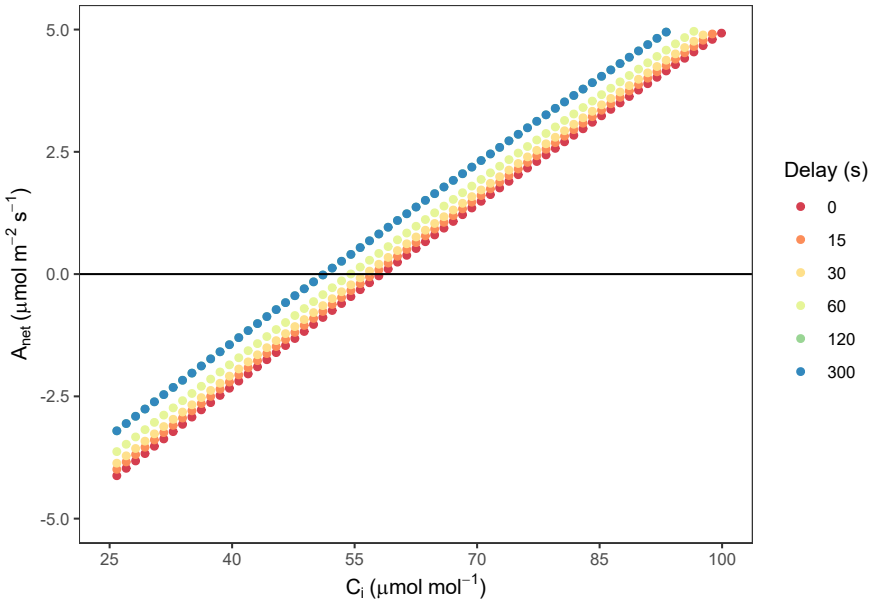


图 10.7: Anet VS. Ci

```
# 表观光合与 Cc 作图
AappCc <- ggplot(data, aes(x = Ccfull, y = Aapp, colour = Delay)) +
  geom_point() +
  labs(x = expression(C[c] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[apparent] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
```

```
  labs(colour = 'Delay (s)') +
  scale_x_continuous(limits = c(25, 75),
                     breaks = c(25, 35, 45, 55, 65, 75)) +
  scale_y_continuous(limits = c(-5, 5)) +
  scale_colour_brewer(palette = 'Spectral') +
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AappCc
```

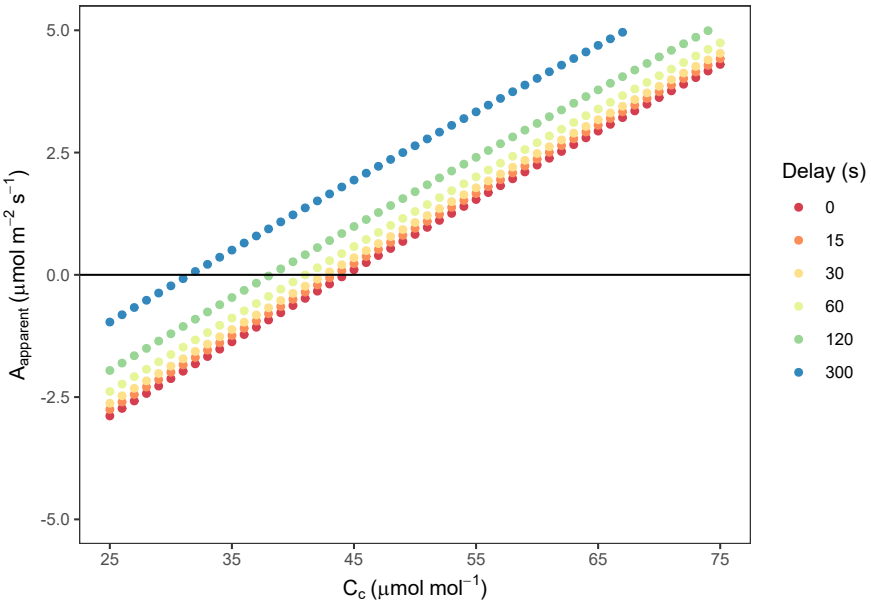


图 10.8: Aapparent VS. Cc

```
# 表观与 Ci 作图
AappCi <- ggplot(data, aes(x = Cifull, y = Aapp, colour = Delay)) +
  geom_point() +
```

```
  labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[apparent] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
  labs(colour = 'Delay (s)') +
  scale_x_continuous(limits = c(25, 75),
                     breaks = c(25, 35, 45, 55, 65, 75)) +
  scale_y_continuous(limits = c(-5, 5)) +
  scale_colour_brewer(palette = 'Spectral') +
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AappCi
```

10.9.5 补偿点计算

计算不同的光呼吸时间延迟下的补偿点（基于 Ci）：

```
# 对于基于 Ci 的数据，仅采用 ci<100 时的数据
dataCi <- data[data$Cifull < 100,]
dataCi0 <- dataCi[dataCi$Delay == "0",]
dataCi15 <- dataCi[dataCi$Delay == "15",]
dataCi30 <- dataCi[dataCi$Delay == "30",]
dataCi60 <- dataCi[dataCi$Delay == "60",]
dataCi120 <- dataCi[dataCi$Delay == "120",]
```

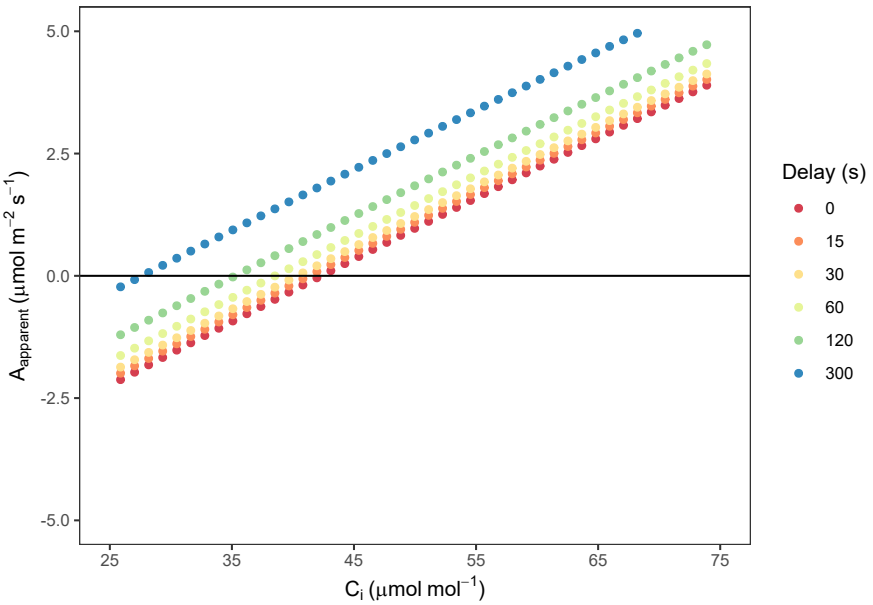


图 10.9: Aapparent VS. Ci

```
dataCi300 <- dataCi[dataCi$Delay == "300",]

# 光呼吸无延迟时的计算，线性拟合
m1 <- lm(dataCi0$Anet ~ dataCi0$Cifull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCi0$Anet ~ dataCi0$Cifull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.13165 -0.04518  0.01711  0.05408  0.06699
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -7.2248105  0.0199353  -362.4   <2e-16 ***
## dataCi0$Cifull  0.1228632  0.0003089   397.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06081 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.582e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
# 补偿点为截距比斜率（纵坐标为零）
Gamma0 <- -m1$coefficients[1] / m1$coefficients[2]

# 光呼吸延时 15s
m2 <- lm(dataCi15$Anet ~ dataCi15$Cifull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCi15$Anet ~ dataCi15$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13119 -0.04502  0.01705  0.05389  0.06676
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -7.0873326  0.0198665  -356.7   <2e-16 ***
## dataCi15$Cifull  0.1225900  0.0003078   398.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.0606 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.586e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma15 <- -m2$coefficients[1] / m2$coefficients[2]

# 光呼吸延时 30s
m3 <- lm(dataCi30$Anet ~ dataCi30$Cifull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCi30$Anet ~ dataCi30$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13076 -0.04488  0.01699  0.05372  0.06654
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -6.9553180  0.0198028  -351.2   <2e-16 ***
## dataCi30$Cifull  0.1223321  0.0003068   398.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0604 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.59e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma30 <- -m3$coefficients[1] / m3$coefficients[2]

# 光呼吸延时 60s
m4 <- lm(dataCi60$Anet ~ dataCi60$Cifull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCi60$Anet ~ dataCi60$Cifull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.13001 -0.04462  0.01690  0.05341  0.06616
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -6.706429   0.019689  -340.6   <2e-16 ***
## dataCi60$Cifull  0.121858   0.000305   399.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06006 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.596e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma60 <- -m4$coefficients[1] / m4$coefficients[2]

# 光呼吸延时 120s
m5 <- lm(dataCi120$Anet ~ dataCi120$Cifull)
summary(m5)
```

```
##
## Call:
## lm(formula = dataCi120$Anet ~ dataCi120$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.12879 -0.04421  0.01673  0.05292  0.06555
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -6.2616980  0.0195062  -321.0   <2e-16 ***
## dataCi120$Cifull  0.1210486  0.0003022   400.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0595 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.604e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma120 <- -m5$coefficients[1] / m5$coefficients[2]

# 光呼吸延时 300s
m6 <- lm(dataCi300$Anet ~ dataCi300$Cifull)
summary(m6)
```

```
##
## Call:
## lm(formula = dataCi300$Anet ~ dataCi300$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.12879 -0.04421  0.01673  0.05292  0.06555
```

```
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -6.2616980  0.0195062  -321.0   <2e-16 ***
## dataCi300$Cifull  0.1210486  0.0003022   400.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0595 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.604e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma300 <- -m6$coefficients[1] / m6$coefficients[2]
```

构造数据并作图

```
GammaCi <- c(Gamma0, Gamma15, Gamma30, Gamma60, Gamma120, Gamma300)

ints <-
  c(
    m1$coefficients[1],
    m2$coefficients[1],
    m3$coefficients[1],
    m4$coefficients[1],
    m5$coefficients[1],
    m6$coefficients[1]
  )
slps <-
  c(
    0,
    m2$coefficients[2] - m1$coefficients[2],
```

```
    m3$coefficients[2] - m1$coefficients[2],
    m4$coefficients[2] - m1$coefficients[2],
    m5$coefficients[2] - m1$coefficients[2],
    m6$coefficients[2] - m1$coefficients[2]
  )
dels <- c(0, 15, 30, 60, 120, 300)
summary(lm(ints ~ dels))
```

```
##
## Call:
## lm(formula = ints ~ dels)
##
## Residuals:
##   (Intercept) (Intercept).1 (Intercept).2 (Intercept).3 (Intercept).4
##      -0.20351      -0.11262      -0.02719       0.12853       0.38691
## (Intercept).5
##      -0.17212
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -7.0213001  0.1343090 -52.277 8.01e-07 ***
## dels         0.0031057  0.0009959   3.119   0.0356 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2503 on 4 degrees of freedom
## Multiple R-squared:  0.7086, Adjusted R-squared:  0.6357
## F-statistic: 9.725 on 1 and 4 DF,  p-value: 0.03558
```

```
plot(ints ~ dels)
```

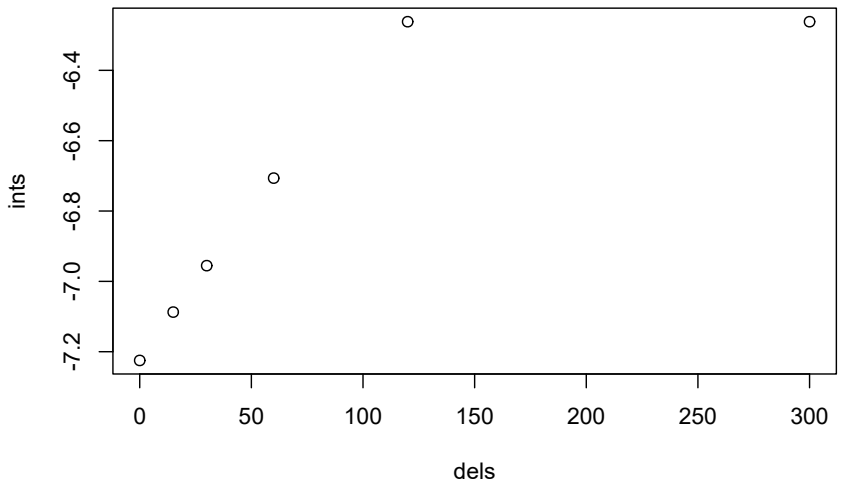


图 10.10: 基于 Ci 的不同延时下的截距

```
plot(slps ~ dels)
```

```
summary(lm(slps ~ dels - 1))
```

```
##
## Call:
## lm(formula = slps ~ dels - 1)
##
## Residuals:
##                   dataCi15$Cifull  dataCi30$Cifull  dataCi60$Cifull
##       -2.711e-19       -1.574e-04       -2.995e-04       -5.424e-04
## dataCi120$Cifull dataCi300$Cifull
##       -8.881e-04        5.015e-04
##
```

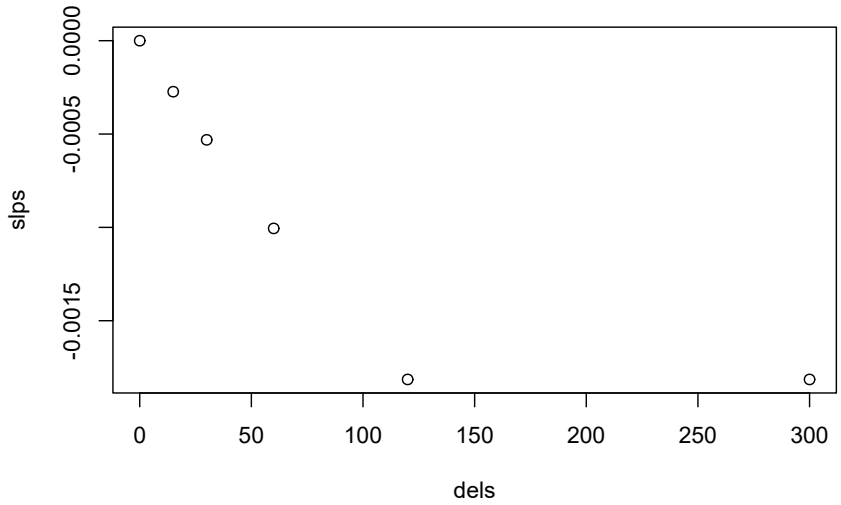


图 10.11: 基于 Ci 的不同延时下的斜率变化

```
## Coefficients:
##       Estimate Std. Error t value Pr(>|t|)
## dels -7.72e-06   1.63e-06  -4.738  0.00516 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0005383 on 5 degrees of freedom
## Multiple R-squared:  0.8178, Adjusted R-squared:  0.7814
## F-statistic: 22.44 on 1 and 5 DF,  p-value: 0.005161
```

基于 Cc 的补偿点计算结果:

```
# 仅使用 Cc < 75 的数据点拟合，过程同 ci
dataCc <- data[data$Ccfull < 75, ]
dataCc0 <- dataCc[dataCc$Delay == "0", ]
```

```
dataCc15 <- dataCc[dataCc$Delay == "15", ]
dataCc30 <- dataCc[dataCc$Delay == "30", ]
dataCc60 <- dataCc[dataCc$Delay == "60", ]
dataCc120 <- dataCc[dataCc$Delay == "120", ]
dataCc300 <- dataCc[dataCc$Delay == "300", ]

# 无延迟数据
m1 <- lm(dataCc0$Anet ~ dataCc0$Ccfull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCc0$Anet ~ dataCc0$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075648 -0.025444  0.009857  0.031706  0.039431
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -8.4062011  0.0181852  -462.3   <2e-16 ***
## dataCc0$Ccfull  0.1438710  0.0003527   407.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03599 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.664e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma0 <- -m1$coefficients[1] / m1$coefficients[2]
```

```
# 延时 15s 数据
m2 <- lm(dataCc15$Anet ~ dataCc15$Ccfull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCc15$Anet ~ dataCc15$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075393 -0.025359  0.009824  0.031599  0.039299
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -8.2659145  0.0181243  -456.1   <2e-16 ***
## dataCc15$Ccfull  0.1435470  0.0003515   408.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03587 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.668e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma15 <- -m2$coefficients[1] / m2$coefficients[2]

# 延时 30s 数据
m3 <- lm(dataCc30$Anet ~ dataCc30$Ccfull)
summary(m3)
```

```
##
## Call:
```

```
## lm(formula = dataCc30$Anet ~ dataCc30$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075158 -0.025280  0.009794  0.031501  0.039177
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -8.1312578  0.0180681  -450.0   <2e-16 ***
## dataCc30$Ccfull  0.1432414  0.0003504   408.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03576 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.671e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma30 <- -m3$coefficients[1] / m3$coefficients[2]

# 延时 60s 数据
m4 <- lm(dataCc60$Anet ~ dataCc60$Ccfull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCc60$Anet ~ dataCc60$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.074737 -0.025139  0.009739  0.031325  0.038959
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -7.8775340  0.0179675  -438.4   <2e-16 ***
## dataCc60$Ccfull  0.1426797  0.0003485   409.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03556 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.676e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma60 <- -m4$coefficients[1] / m4$coefficients[2]

# 延时 120s 数据
m5 <- lm(dataCc120$Anet ~ dataCc120$Ccfull)
summary(m5)
```

```
##
## Call:
## lm(formula = dataCc120$Anet ~ dataCc120$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.074060 -0.024912  0.009651  0.031042  0.038607
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -7.4246412  0.0178052  -417.0   <2e-16 ***
## dataCc120$Ccfull  0.1417238  0.0003453   410.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03524 on 48 degrees of freedom
```

```
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.684e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma120 <- -m5$coefficients[1] / m5$coefficients[2]

# 延时 300s 数据
m6 <- lm(dataCc300$Anet ~ dataCc300$Ccfull)
summary(m6)
```

```
##
## Call:
## lm(formula = dataCc300$Anet ~ dataCc300$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.074060 -0.024912  0.009651  0.031042  0.038607
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -7.4246412  0.0178052  -417.0   <2e-16 ***
## dataCc300$Ccfull  0.1417238  0.0003453   410.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03524 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.684e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma300 <- -m6$coefficients[1] / m6$coefficients[2]
```

```
GammaCc <- c(Gamma0, Gamma15, Gamma30, Gamma60, Gamma120, Gamma300)

ints <-
  c(
    m1$coefficients[1],
    m2$coefficients[1],
    m3$coefficients[1],
    m4$coefficients[1],
    m5$coefficients[1],
    m6$coefficients[1]
  )
dels <- c(0, 15, 30, 60, 120, 300)
plot(ints ~ dels)
```

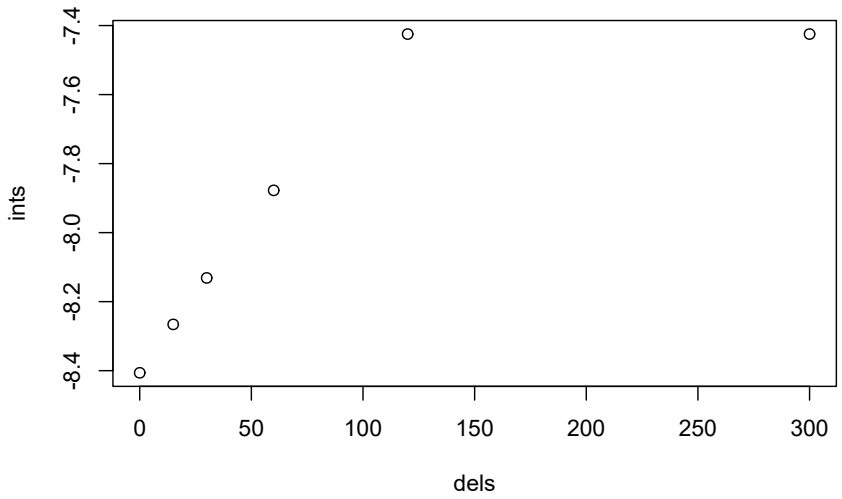


图 10.12: 基于 Cc 的不同延时下的时间

```
summary(lm(ints ~ dels))
```

```
##
## Call:
## lm(formula = ints ~ dels)
##
## Residuals:
##   (Intercept) (Intercept).1 (Intercept).2 (Intercept).3 (Intercept).4
##      -0.20760      -0.11478      -0.02759       0.13119       0.39421
## (Intercept).5
##      -0.17542
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.198602   0.136891 -59.891 4.65e-07 ***
## dels         0.003165   0.001015   3.118   0.0356 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2552 on 4 degrees of freedom
## Multiple R-squared:  0.7085, Adjusted R-squared:  0.6356
## F-statistic:  9.72 on 1 and 4 DF,  p-value: 0.03561
```

```
#GammaStar
# For Ci-based estimates, only use Ci < 100
dataCi <- data[data$Cifull < 100,]
dataCi0 <- dataCi[dataCi$Delay == "0",]
dataCi15 <- dataCi[dataCi$Delay == "15",]
dataCi30 <- dataCi[dataCi$Delay == "30",]
dataCi60 <- dataCi[dataCi$Delay == "60",]
dataCi120 <- dataCi[dataCi$Delay == "120",]
```

```
dataCi300 <- dataCi[dataCi$Delay == "300",]
m1 <- lm(dataCi0$Aapp ~ dataCi0$Cifull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCi0$Aapp ~ dataCi0$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13165 -0.04518  0.01711  0.05408  0.06699
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -5.2248105  0.0199353  -262.1   <2e-16 ***
## dataCi0$Cifull  0.1228632  0.0003089   397.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06081 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.582e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma0 <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCi15$Aapp ~ dataCi15$Cifull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCi15$Aapp ~ dataCi15$Cifull)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13119 -0.04502  0.01705  0.05389  0.06676
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -5.0873326  0.0198665  -256.1   <2e-16 ***
## dataCi15$Cifull  0.1225900  0.0003078   398.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0606 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.586e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma15 <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCi30$Aapp ~ dataCi30$Cifull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCi30$Aapp ~ dataCi30$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13076 -0.04488  0.01699  0.05372  0.06654
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -4.9553180  0.0198028  -250.2   <2e-16 ***
## dataCi30$Cifull  0.1223321  0.0003068   398.7   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0604 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.59e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma30 <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCi60$Aapp ~ dataCi60$Cifull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCi60$Aapp ~ dataCi60$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13001 -0.04462  0.01690  0.05341  0.06616
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -4.706429   0.019689  -239.0   <2e-16 ***
## dataCi60$Cifull  0.121858   0.000305   399.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06006 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.596e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma60 <- -m4$coefficients[1] / m4$coefficients[2]
m5 <- lm(dataCi120$Aapp ~ dataCi120$Cifull)
summary(m5)
```

```
##
## Call:
## lm(formula = dataCi120$Aapp ~ dataCi120$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.12879 -0.04421  0.01673  0.05292  0.06555
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -4.2616980  0.0195062  -218.5   <2e-16 ***
## dataCi120$Cifull  0.1210486  0.0003022   400.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0595 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.604e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma120 <- -m5$coefficients[1] / m5$coefficients[2]
m6 <- lm(dataCi300$Aapp ~ dataCi300$Cifull)
summary(m6)
```

```
##
## Call:
## lm(formula = dataCi300$Aapp ~ dataCi300$Cifull)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.12662 -0.04346  0.01645  0.05202  0.06444
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -3.2402030  0.0191773  -169.0   <2e-16 ***
## dataCi300$Cifull  0.1193852  0.0002971   401.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05849 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.615e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma300 <- -m6$coefficients[1] / m6$coefficients[2]

GammastarCi <-
  c(Gamma0, Gamma15, Gamma30, Gamma60, Gamma120, Gamma300)

# For Cc-based estimates, only use Cc < 75
dataCc <- data[data$Ccfull < 75,]
dataCc0 <- dataCc[dataCc$Delay == "0",]
dataCc15 <- dataCc[dataCc$Delay == "15",]
dataCc30 <- dataCc[dataCc$Delay == "30",]
dataCc60 <- dataCc[dataCc$Delay == "60",]
dataCc120 <- dataCc[dataCc$Delay == "120",]
dataCc300 <- dataCc[dataCc$Delay == "300",]
m1 <- lm(dataCc0$Aapp ~ dataCc0$Ccfull)
summary(m1)
```

```
##
```

```
## Call:
## lm(formula = dataCc0$Aapp ~ dataCc0$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075648 -0.025444  0.009857  0.031706  0.039431
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -6.4062011  0.0181852  -352.3   <2e-16 ***
## dataCc0$Ccfull  0.1438710  0.0003527   407.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03599 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.664e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma0 <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCc15$Aapp ~ dataCc15$Ccfull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCc15$Aapp ~ dataCc15$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075393 -0.025359  0.009824  0.031599  0.039299
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)     -6.2659145  0.0181243  -345.7   <2e-16 ***
## dataCc15$Ccfull  0.1435470  0.0003515   408.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03587 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.668e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma15 <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCc30$Aapp ~ dataCc30$Ccfull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCc30$Aapp ~ dataCc30$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075158 -0.025280  0.009794  0.031501  0.039177
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -6.1312578  0.0180681  -339.3   <2e-16 ***
## dataCc30$Ccfull  0.1432414  0.0003504   408.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03576 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.671e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma30 <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCc60$Aapp ~ dataCc60$Ccfull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCc60$Aapp ~ dataCc60$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.074737 -0.025139  0.009739  0.031325  0.038959
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -5.8775340  0.0179675  -327.1   <2e-16 ***
## dataCc60$Ccfull  0.1426797  0.0003485   409.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03556 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.676e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma60 <- -m4$coefficients[1] / m4$coefficients[2]
m5 <- lm(dataCc120$Aapp ~ dataCc120$Ccfull)
summary(m5)
```

```
##
## Call:
## lm(formula = dataCc120$Aapp ~ dataCc120$Ccfull)
##
```

```
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.074060 -0.024912  0.009651  0.031042  0.038607
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -5.4246412  0.0178052  -304.7   <2e-16 ***
## dataCc120$Ccfull  0.1417238  0.0003453   410.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03524 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.684e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma120 <- -m5$coefficients[1] / m5$coefficients[2]
m6 <- lm(dataCc300$Aapp ~ dataCc300$Ccfull)
summary(m6)
```

```
##
## Call:
## lm(formula = dataCc300$Aapp ~ dataCc300$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.072835 -0.024500  0.009492  0.030529  0.037969
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -4.3867121  0.0175110  -250.5   <2e-16 ***
## dataCc300$Ccfull  0.1397661  0.0003396   411.5   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03466 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.694e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gamma300 <- -m6$coefficients[1] / m6$coefficients[2]

GammastarCc <-
  c(Gamma0, Gamma15, Gamma30, Gamma60, Gamma120, Gamma300)

Delay2 <- c("0", "15", "30", "60", "120", "300")
PRcomps <-
  as.data.frame(cbind(Delay2, GammaCc, GammaCi, GammastarCc, GammastarCi))
write.csv(PRcomps, "./data/PRcomps.csv")

ints <-
  c(
    m1$coefficients[1],
    m2$coefficients[1],
    m3$coefficients[1],
    m4$coefficients[1],
    m5$coefficients[1],
    m6$coefficients[1]
  )
dels <- c(0, 15, 30, 60, 120, 300)
summary(lm(ints ~ dels))
```

```
##
## Call:
## lm(formula = ints ~ dels)
##
```

```
## Residuals:
##   (Intercept) (Intercept).1 (Intercept).2 (Intercept).3 (Intercept).4
##    -0.0751633    -0.0347043     0.0001247     0.0541933     0.1077758
## (Intercept).5
##    -0.0522262
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.331038   0.041673 -151.92 1.13e-08 ***
## dels         0.006655   0.000309   21.54 2.75e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07768 on 4 degrees of freedom
## Multiple R-squared:  0.9915, Adjusted R-squared:  0.9893
## F-statistic: 463.9 on 1 and 4 DF,  p-value: 2.749e-05
```

```
plot(ints ~ dels)
```

10.9.6 无光呼吸酶失活模块

该部分内容是在测量 ACi 曲线时检测 Rubisco 失活的影响 – 从激活状态的变化导致了多少的偏移？

10.9.6.1 数据构造

基于文献，假定 CO_2 从 400 ppm 降低至 5 ppm 时，激活率从 100% 降低至 80%。

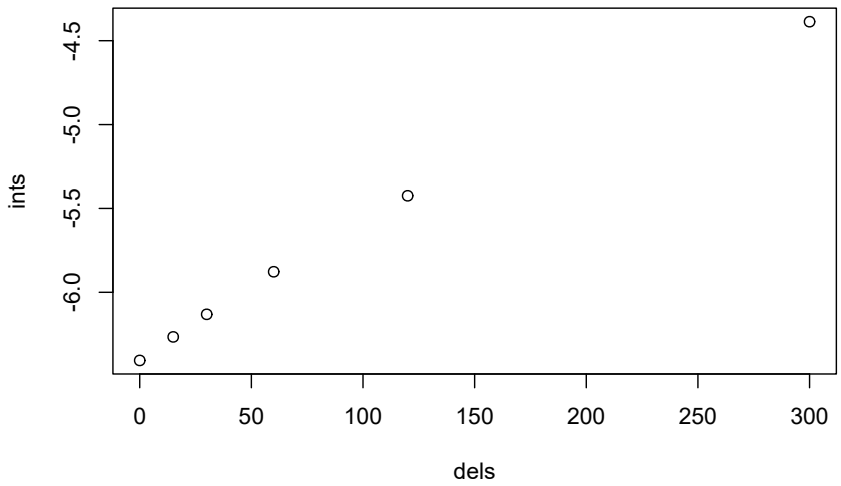


图 10.13: 基于 Cc 的不同延时下的截距

```
#Assume that Rubisco activation state drops from 100% at 400 ppm to 80% at 5 ppm Cr (

# 不同的 cr 对应了不同的 cc 浓度，
# 此为 cc 的变化范围（cr 从 400 降低至 5）
ccslope <- c(25, 297)
# 酶的激活率变化
raslope <- c(0.80, 1.00)
# 得到 cc 变化对应 rubisco 激活率变化的关系
ram1 <- lm(raslope ~ ccslope)
raslope <- coef(ram1)[2]
raint <- coef(ram1)[1]

# 根据公式计算酶部分失活后各个参数
vora1 <- (raslope * Cc + raint) * Vomax * O2 / (O2 + Ko * (1 + Cc / Kc))
vcra1 <- (raslope * Cc + raint) * Vcmax * (Cc) / (Cc + Kco)
```

```
Ara1 <- vcra1 - 0.5 * vora1 - R
Aapparentra1 <- vcra1 - 0.5 * vora1
Cira1 <- Ara1 / gm + Cc
Cbra1 <- Ara1 / gsw + Cira1
Crra1 <- Ara1 / BLC + Cbra1

# 失活后换算为分钟的变化斜率
Counter <- as.numeric(c(1:length(Cc)))
RateCr1model <- lm(Crra1 ~ Counter)
RateCr1 <- coef(RateCr1model)[2] * 60
RateCb1model <- lm(Cbra1 ~ Counter)
RateCb1 <- coef(RateCb1model)[2] * 60
RateCi1model <- lm(Cira1 ~ Counter)
RateCi1 <- coef(RateCi1model)[2] * 60
RateCcmodel <- lm(Cc ~ Counter)
RateCc <- coef(RateCcmodel)[2] * 60

# 假定在 5ppm 时下降为 40%
ccslope2 <- c(25, 297)
raslope2 <- c(0.40, 1.00)
ram2 <- lm(raslope2 ~ ccslope2)
raslope2 <- coef(ram2)[2]
raint2 <- coef(ram2)[1]

vora2 <-
  (raslope2 * Cc + raint2) * Vomax * O2 / (O2 + Ko * (1 + Cc / Kc))
vcra2 <- (raslope2 * Cc + raint2) * Vcmax * (Cc) / (Cc + Kco)
Ara2 <- vcra2 - 0.5 * vora2 - R
Aapparentra2 <- vcra2 - 0.5 * vora2
Cira2 <- Ara2 / gm + Cc #umol mol-1
Cbra2 <- Ara2 / gsw + Cira2 #umol mol-1
Crra2 <- Ara2 / BLC + Cbra2 #umol mol-1
```

```
Counter <- as.numeric(c(1:length(Cc)))
RateCr2model <- lm(Crra2 ~ Counter)
RateCr2 <- coef(RateCr2model)[2] * 60 #umol mol-1 min-1
RateCb2model <- lm(Cbra2 ~ Counter)
RateCb2 <- coef(RateCb2model)[2] * 60 #umol mol-1 min-1
RateCi2model <- lm(Cira2 ~ Counter)
RateCi2 <- coef(RateCi2model)[2] * 60 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc <- coef(RateCcmodel)[2] * 60 #umol mol-1 min-1

# 假定在 5ppm 时下降为 20%
ccslope3 <- c(25, 297)
raslope3 <- c(0.20, 1.00)
ram3 <- lm(raslope3 ~ ccslope3)
raslope3 <- coef(ram3)[2]
raint3 <- coef(ram3)[1]

vora3 <-
  (raslope3 * Cc + raint3) * Vomax * O2 / (O2 + Ko * (1 + Cc / Kc))
vcra3 <- (raslope3 * Cc + raint3) * Vcmax * (Cc) / (Cc + Kco)
Ara3 <- vcra3 - 0.5 * vora3 - R
Aapparentra3 <- vcra3 - 0.5 * vora3
Cira3 <- Ara3 / gm + Cc
Cbra3 <- Ara3 / gsw + Cira3
Crra3 <- Ara3 / BLC + Cbra3
Counter <- as.numeric(c(1:length(Cc)))
RateCr3model <- lm(Crra3 ~ Counter)
RateCr3 <- coef(RateCr3model)[2] * 60
RateCb3model <- lm(Cbra3 ~ Counter)
RateCb3 <- coef(RateCb3model)[2] * 60
RateCi3model <- lm(Cira3 ~ Counter)
RateCi3 <- coef(RateCi3model)[2] * 60
```

```
RateCcmodel <- lm(Cc ~ Counter)
RateCc <- coef(RateCcmodel)[2] * 60

Anet <- c(A, Ara1, Ara2, Ara3)
Aapp <- c(Aapparent, Aapparentra1, Aapparentra2, Aapparentra3)
Ccfull <- rep(Cc, 4)
Cifull <- c(Ci, Cira1, Cira2, Cira3)
Deactivation <-
  c(
    rep("None", length(A)),
    rep("Low", length(Ara1)),
    rep("Medium", length(Ara2)),
    rep("High", length(Ara3))
  )
RASdata <-
  as.data.frame(cbind(Anet, Aapp, Ccfull, Cifull, Deactivation))
write.csv(RASdata, "./data/RASdata.csv")
```

10.9.7 酶失活作图

```
data <- read.csv("./data/RASdata.csv")
data$Ccfull <- as.numeric(data$Ccfull)
data$Deactivation <- as.factor(data$Deactivation)

AnetCc <-
  ggplot(data, aes(x = Ccfull, y = Anet, colour = Deactivation)) +
  geom_point() +
  labs(x = expression(C[c] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
```

```
  y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
  labs(colour = 'Deactivation') +
  scale_x_continuous(limits = c(25, 100),
                     breaks = c(25, 40, 55, 70, 85, 100)) +
  scale_y_continuous(limits = c(-5, 5)) +
  scale_colour_brewer(palette = 'Spectral') +
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AnetCc
```

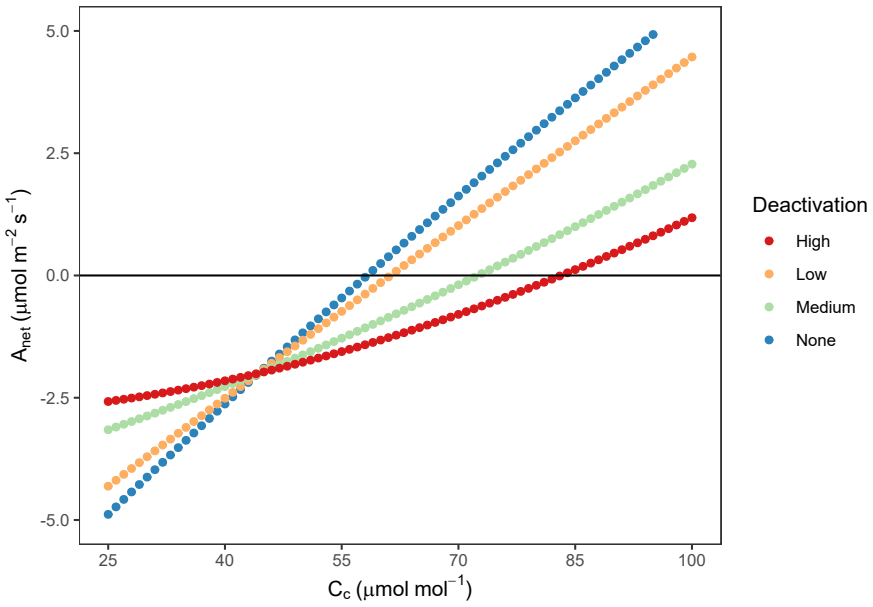


图 10.14: Rubisco 不同失活程度时 Anet VS Cc

```
AnetCi <-
  ggplot(data, aes(x = Cifull, y = Anet, colour = Deactivation)) +
  geom_point() +
  labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
  labs(colour = 'Deactivation') +
  scale_x_continuous(limits = c(25, 100),
                     breaks = c(25, 40, 55, 70, 85, 100)) +
  scale_y_continuous(limits = c(-5, 5)) +
  scale_colour_brewer(palette = 'Spectral') +
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AnetCi
```

```
AappCc <-
  ggplot(data, aes(x = Ccfull, y = Aapp, colour = Deactivation)) +
  geom_point() +
  labs(x = expression(C[c] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[apparent] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
```

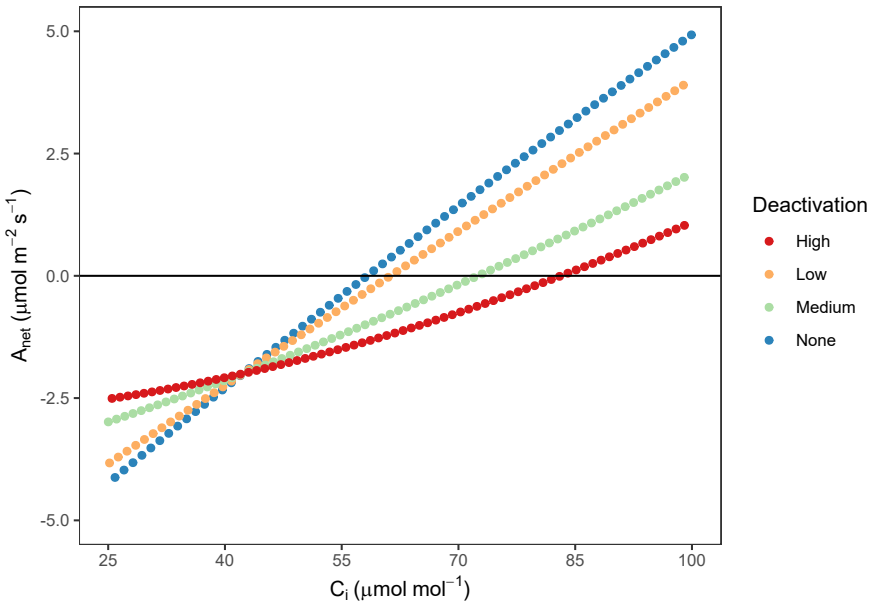


图 10.15: Rubisco 不同失活程度时 Anet VS Ci

```
    -1
  } * ")")) +
  labs(colour = 'Deactivation') +
  scale_x_continuous(limits = c(25, 75),
                     breaks = c(25, 35, 45, 55, 65, 75)) +
  scale_y_continuous(limits = c(-5, 5)) +
  scale_colour_brewer(palette = 'Spectral') +
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AappCc
```

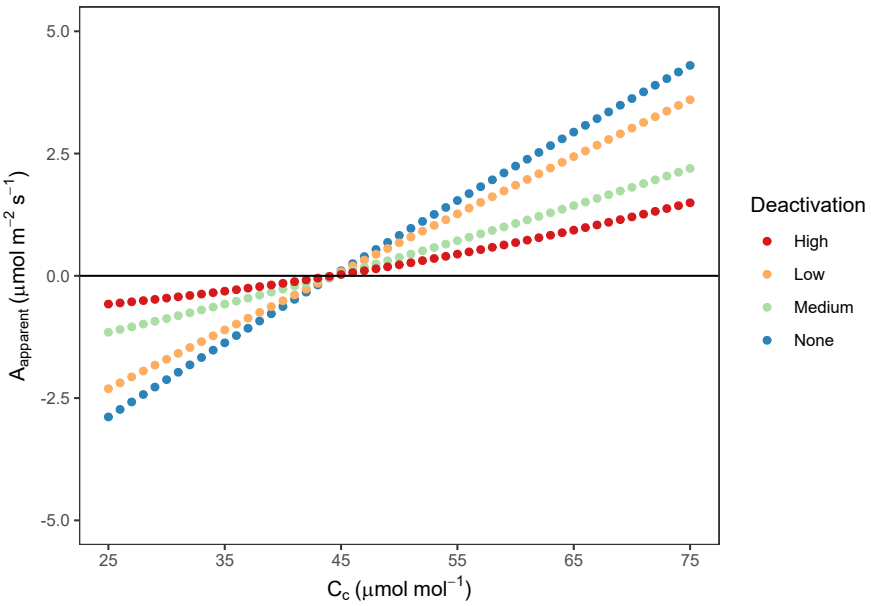


图 10.16: Rubisco 不同失活程度时 Aapp VS Cc

```
AappCi <-
  ggplot(data, aes(x = Cifull, y = Aapp, colour = Deactivation)) +
  geom_point() +
  labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[apparent] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
  labs(colour = 'Deactivation') +
  scale_x_continuous(limits = c(25, 75),
                     breaks = c(25, 35, 45, 55, 65, 75)) +
  scale_y_continuous(limits = c(-5, 5)) +
```

```
  scale_colour_brewer(palette = 'Spectral') +
  geom_hline(yintercept = 0) +
  theme_bw() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank())
AappCi
```

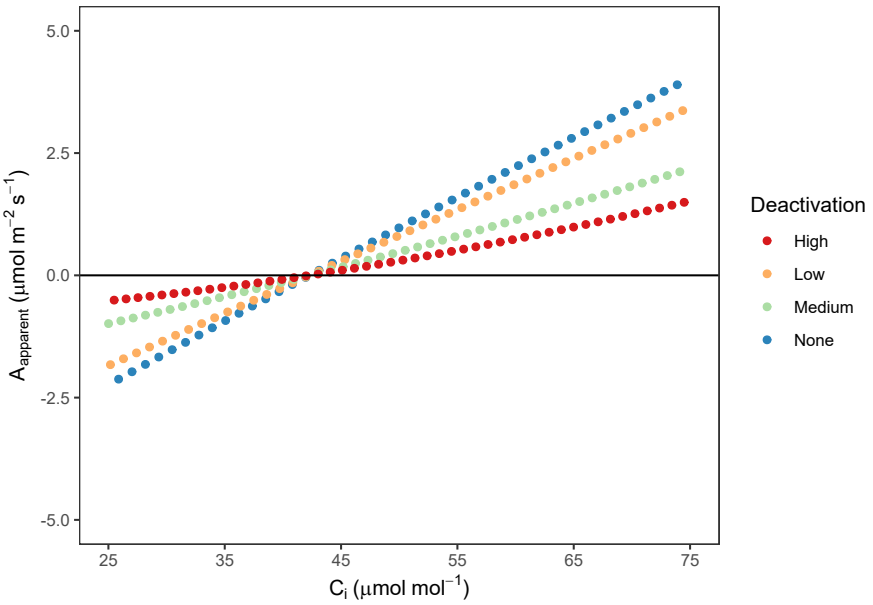


图 10.17: Rubisco 不同失活程度时 Aapp VS Ci

10.9.8 不同失活程度下补偿点计算

此部分内容同未失活状态相似，不在额外介绍，可参考 **??** 内容。

```
#Gamma
# For Ci-based estimates, only use Ci < 100
dataCi <- data[data$Cifull < 100, ]
```

```
dataCinone <- dataCi[dataCi$Deactivation == "None", ]
dataCilow <- dataCi[dataCi$Deactivation == "Low", ]
dataCimedium <- dataCi[dataCi$Deactivation == "Medium", ]
dataCihigh <- dataCi[dataCi$Deactivation == "High", ]
m1 <- lm(dataCinone$Anet ~ dataCinone$Cifull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCinone$Anet ~ dataCinone$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13165 -0.04518  0.01711  0.05408  0.06699
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -7.2248105  0.0199353  -362.4   <2e-16 ***
## dataCinone$Cifull  0.1228632  0.0003089   397.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06081 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.582e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gammanone <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCilow$Anet ~ dataCilow$Cifull)
summary(m2)
```

```
##
```

```
## Call:
## lm(formula = dataCilow$Anet ~ dataCilow$Cifull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.034896 -0.011957  0.004499  0.014311  0.017726
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -6.444e+00  5.343e-03   -1206   <2e-16 ***
## dataCilow$Cifull  1.049e-01  8.341e-05    1258   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01609 on 69 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.582e+06 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gammalow <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCimedium$Anet ~ dataCimedium$Cifull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCimedium$Anet ~ dataCimedium$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09168 -0.07338 -0.02281  0.05893  0.18150
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)         -4.8015337  0.0277371  -173.1   <2e-16 ***
## dataCimedium$Cifull  0.0671064  0.0004311   155.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0832 on 71 degrees of freedom
## Multiple R-squared:  0.9971, Adjusted R-squared:  0.997
## F-statistic: 2.423e+04 on 1 and 71 DF,  p-value: < 2.2e-16
```

```
Gammamedium <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCihigh$Anet ~ dataCihigh$Cifull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCihigh$Anet ~ dataCihigh$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.15430 -0.12508 -0.03863  0.10156  0.30621
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -3.9450658  0.0467537  -84.38   <2e-16 ***
## dataCihigh$Cifull  0.0473551  0.0007255   65.27   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1399 on 72 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9831
## F-statistic:  4260 on 1 and 72 DF,  p-value: < 2.2e-16
```

```
Gammahigh <- -m4$coefficients[1] / m4$coefficients[2]

GammaCi <- c(Gammanone, Gammalow, Gammamedium, Gammahigh)

# For Cc-based estimates, only use Cc < 75
dataCc <- data[data$Ccfull < 75, ]
dataCcnone <- dataCc[dataCc$Deactivation == "None", ]
dataCclow <- dataCc[dataCc$Deactivation == "Low", ]
dataCcmedium <- dataCc[dataCc$Deactivation == "Medium", ]
dataCchigh <- dataCc[dataCc$Deactivation == "High", ]
m1 <- lm(dataCcnone$Anet ~ dataCcnone$Ccfull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCcnone$Anet ~ dataCcnone$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075648 -0.025444  0.009857  0.031706  0.039431
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -8.4062011  0.0181852  -462.3   <2e-16 ***
## dataCcnone$Ccfull  0.1438710  0.0003527   407.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03599 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.664e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammanone <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCclow$Anet ~ dataCclow$Ccfull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCclow$Anet ~ dataCclow$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.019617 -0.006598  0.002556  0.008222  0.010225
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -7.243e+00  4.716e-03   -1536   <2e-16 ***
## dataCclow$Ccfull  1.182e-01  9.146e-05    1292   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009333 on 48 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.67e+06 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammalow <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCcmedium$Anet ~ dataCcmedium$Ccfull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCcmedium$Anet ~ dataCcmedium$Ccfull)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04819 -0.03875 -0.01205  0.03109  0.09244
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -4.917283   0.022223  -221.3   <2e-16 ***
## dataCcmedium$Ccfull  0.066832   0.000431   155.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04398 on 48 degrees of freedom
## Multiple R-squared:  0.998,  Adjusted R-squared:  0.998
## F-statistic: 2.404e+04 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammamedium <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCchigh$Anet ~ dataCchigh$Ccfull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCchigh$Anet ~ dataCchigh$Ccfull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.07739 -0.06223 -0.01935  0.04994  0.14847
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -3.7543099  0.0356922 -105.19   <2e-16 ***
## dataCchigh$Ccfull  0.0411528  0.0006922   59.45   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07064 on 48 degrees of freedom
## Multiple R-squared:  0.9866, Adjusted R-squared:  0.9863
## F-statistic:  3534 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammahigh <- -m4$coefficients[1] / m4$coefficients[2]

GammaCc <- c(Gammanone, Gammalow, Gammamedium, Gammahigh)

#GammaStar
# For Ci-based estimates, only use Ci < 100
dataCi <- data[data$Cifull < 100, ]
dataCinone <- dataCi[dataCi$Deactivation == "None", ]
dataCilow <- dataCi[dataCi$Deactivation == "Low", ]
dataCimedium <- dataCi[dataCi$Deactivation == "Medium", ]
dataCihigh <- dataCi[dataCi$Deactivation == "High", ]
m1 <- lm(dataCinone$Aapp ~ dataCinone$Cifull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCinone$Aapp ~ dataCinone$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13165 -0.04518  0.01711  0.05408  0.06699
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -5.2248105  0.0199353  -262.1   <2e-16 ***
## dataCinone$Cifull  0.1228632  0.0003089   397.8   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06081 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.582e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gammastarnone <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCilow$Aapp ~ dataCilow$Cifull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCilow$Aapp ~ dataCilow$Cifull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.034896 -0.011957  0.004499  0.014311  0.017726
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -4.444e+00  5.343e-03  -831.7   <2e-16 ***
## dataCilow$Cifull  1.049e-01  8.341e-05  1257.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01609 on 69 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.582e+06 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gammastarlow <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCimedium$Aapp ~ dataCimedium$Cifull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCimedium$Aapp ~ dataCimedium$Cifull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.09168 -0.07338 -0.02281  0.05893  0.18150
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.8015337  0.0277371  -101.0   <2e-16 ***
## dataCimedium$Cifull  0.0671064  0.0004311   155.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0832 on 71 degrees of freedom
## Multiple R-squared:  0.9971, Adjusted R-squared:  0.997
## F-statistic: 2.423e+04 on 1 and 71 DF,  p-value: < 2.2e-16
```

```
Gammastarmedium <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCihigh$Aapp ~ dataCihigh$Cifull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCihigh$Aapp ~ dataCihigh$Cifull)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.15430 -0.12508 -0.03863  0.10156  0.30621
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.9450658  0.0467537  -41.60   <2e-16 ***
## dataCihigh$Cifull  0.0473551  0.0007255   65.27   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1399 on 72 degrees of freedom
## Multiple R-squared:  0.9834, Adjusted R-squared:  0.9831
## F-statistic:  4260 on 1 and 72 DF,  p-value: < 2.2e-16
```

```
Gammastarhigh <- -m4$coefficients[1] / m4$coefficients[2]

GammastarCi <-
  c(Gammastarnone, Gammastarlow, Gammastarmedium, Gammastarhigh)

# For Cc-based estimates, only use Cc < 75
dataCc <- data[data$Ccfull < 75, ]
dataCcnone <- dataCc[dataCc$Deactivation == "None", ]
dataCclow <- dataCc[dataCc$Deactivation == "Low", ]
dataCcmedium <- dataCc[dataCc$Deactivation == "Medium", ]
dataCchigh <- dataCc[dataCc$Deactivation == "High", ]
m1 <- lm(dataCcnone$Aapp ~ dataCcnone$Ccfull)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCcnone$Aapp ~ dataCcnone$Ccfull)
```

```
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.075648 -0.025444  0.009857  0.031706  0.039431
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -6.4062011  0.0181852  -352.3   <2e-16 ***
## dataCcnone$Ccfull  0.1438710  0.0003527   407.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03599 on 48 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 1.664e+05 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammastarnone <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCclow$Aapp ~ dataCclow$Ccfull)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCclow$Aapp ~ dataCclow$Ccfull)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.019617 -0.006598  0.002556  0.008222  0.010225
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -5.243e+00  4.716e-03   -1112   <2e-16 ***
## dataCclow$Ccfull  1.182e-01  9.146e-05    1292   <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.009333 on 48 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.67e+06 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammastarlow <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCcmedium$Aapp ~ dataCcmedium$Ccfull)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCcmedium$Aapp ~ dataCcmedium$Ccfull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.04819 -0.03875 -0.01205  0.03109  0.09244
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -2.917283   0.022223  -131.3   <2e-16 ***
## dataCcmedium$Ccfull  0.066832   0.000431   155.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04398 on 48 degrees of freedom
## Multiple R-squared:  0.998,  Adjusted R-squared:  0.998
## F-statistic: 2.404e+04 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammastarmedium <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCchigh$Aapp ~ dataCchigh$Ccfull)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCchigh$Aapp ~ dataCchigh$Ccfull)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.07739 -0.06223 -0.01935  0.04994  0.14847
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.7543099  0.0356922  -49.15   <2e-16 ***
## dataCchigh$Ccfull  0.0411528  0.0006922   59.45   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07064 on 48 degrees of freedom
## Multiple R-squared:  0.9866, Adjusted R-squared:  0.9863
## F-statistic:  3534 on 1 and 48 DF,  p-value: < 2.2e-16
```

```
Gammastarhigh <- -m4$coefficients[1] / m4$coefficients[2]

GammastarCc <-
  c(Gammastarnone, Gammastarlow, Gammastarmedium, Gammastarhigh)

Deactivation2 <- c("None", "Low", "Medium", "High")
RAScomps <-
```

```
  as.data.frame(cbind(Deactivation2, GammaCc, GammaCi, GammastarCc, GammastarCi))
write.csv(RAScomps, "./data/RAScomps.csv")
```

10.10 时间延迟的扩散限制

对于扩散限制，下面的内容比较了多速率 RACiR 和标准 ACi 曲线的差别，比较实在有光呼吸和没有光呼吸的两种情况。对于没有扩散限制的表观光合速，采用了已知质量的碱石灰药品，放置于 1.7 ml 的微量离心管内，然后将其置于荧光叶室内部模拟叶片，此时叶室环境控制与其他实验不同，此时不再控制 H2OR。RACiR 测试从 500 到 0 的变化，不同样品的测量是随机的。

下面内容采用了一定的假设，来计算扩散的时间。

```
#Equations from Campbell & Norman, 1998
#We are taking a simple approach to calculating diffusion times.
#Here we make the simplifying assumption that diffusion is pure, planar
#Diffusion, such that:
#gtot = phat * D / deltaZ
#where gtot is total conductance, phat is molar density of air in mol /m^3,
#D is diffusion coefficient in m2/s, deltaZ is pathlength in m
#Since PV = NRT, N/V = P/RT
#T in K, R in J K-1 mol-1, P in Pa

#phat = Patm/(RT)
phat = 100000 / (8.314 * 298.15)

#We also assume a linear pathlength
#Note, if diffusion is nonlinear or nonplanar, it will affect the value determined
#for D from this equation.
```

```
#D = gtot * deltaZ / phat

#Diffusion time, t, varies with D and deltaZ such that:

#t = (deltaZ)^2 / D

#So

#t = (deltaZ)^2 / (gtot * deltaZ / phat)

#If we assume mean diffusion pathlength of 1/2 lamina thickness,
#then Onoda et al. 2011 lamina thicknesses of: median 0.22 mm (0.11 to 0.74 for 95% C
#becomes 0.11 mm (0.055 to 0.37) for estimated deltaZ

#Convert pathlength to m
dZlow <- 0.055 / 1000
dZmedian <- 0.11 / 1000
dZhigh <- 0.37 / 1000
```

下面的内容是对边界层导度和气孔导度等赋值，由此而计算出其他所需要的参数：

```
#Mesophyll Conductance
BLC <- 2 #mol m-2 s-1
gsw <- 0.4 #mol m-2 s-1

# 无限制的叶肉导度，并以此计算 ci 等
gm1 <- 1 #mol m-2 s-1
Ci1 <- A / gm1 + Cc #umol mol-1
Cim1 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
```

```
Cr1 <- A / BLC + Cb1 #umol mol-1
# 根据斜率计算达到 100 ppm min-1 时记录数据的个数
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
# 计算 ci, Cc 等达到 100ppm min-1 时数据的个数
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc1 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
# 总的阻力
res1 <- 1 / gm1 + 1 / BLC + 1 / gsw

# 不同的叶肉导度计算其他参数
gm2 <- 2 #mol m-2 s-1
Ci2 <- A / gm2 + Cc #umol mol-1
Cim2 <- Ci2
Cb2 <- A / gsw + Ci2 #umol mol-1
Cr2 <- A / BLC + Cb2 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr2 ~ Counter)
x2 <- 100 / coef(RateCrmodel)[2]
RateCr2 <- coef(RateCrmodel)[2] * x2 #umol mol-1 min-1
RateCbmodel <- lm(Cb2 ~ Counter)
RateCb2 <- coef(RateCbmodel)[2] * x2 #umol mol-1 min-1
RateCimodel <- lm(Ci2 ~ Counter)
RateCi2 <- coef(RateCimodel)[2] * x2 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc2 <- coef(RateCcmodel)[2] * x2 #umol mol-1 min-1
```

```
res2 <- 1 / gm2 + 1 / BLC + 1 / gsw

# 再次计算不同导度下的数值
gm4 <- 4 #mol m-2 s-1
Ci4 <- A / gm4 + Cc #umol mol-1
Cim4 <- Ci4
Cb4 <- A / gsw + Ci4 #umol mol-1
Cr4 <- A / BLC + Cb4 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr4 ~ Counter)
x4 <- 100 / coef(RateCrmodel)[2]
RateCr4 <- coef(RateCrmodel)[2] * x4 #umol mol-1 min-1
RateCbmodel <- lm(Cb4 ~ Counter)
RateCb4 <- coef(RateCbmodel)[2] * x4 #umol mol-1 min-1
RateCimodel <- lm(Ci4 ~ Counter)
RateCi4 <- coef(RateCimodel)[2] * x4 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc4 <- coef(RateCcmodel)[2] * x4 #umol mol-1 min-1
res4 <- 1 / gm4 + 1 / BLC + 1 / gsw

# 再次计算不同导度下的数值
gm05 <- 0.5 #mol m-2 s-1
Ci05 <- A / gm05 + Cc #umol mol-1
Cim05 <- Ci05
Cb05 <- A / gsw + Ci05 #umol mol-1
Cr05 <- A / BLC + Cb05 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr05 ~ Counter)
x05 <- 100 / coef(RateCrmodel)[2]
RateCr05 <- coef(RateCrmodel)[2] * x05 #umol mol-1 min-1
RateCbmodel <- lm(Cb05 ~ Counter)
RateCb05 <- coef(RateCbmodel)[2] * x05 #umol mol-1 min-1
```

```
RateCimodel <- lm(Ci05 ~ Counter)
RateCi05 <- coef(RateCimodel)[2] * x05 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc05 <- coef(RateCcmodel)[2] * x05 #umol mol-1 min-1
res05 <- 1 / gm05 + 1 / BLC + 1 / gsw

# 正常的叶肉导度数据计算其他参数
gm025 <- 0.25 #mol m-2 s-1
Ci025 <- A / gm025 + Cc #umol mol-1
Cim025 <- Ci025
Cb025 <- A / gsw + Ci025 #umol mol-1
Cr025 <- A / BLC + Cb025 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr025 ~ Counter)
x025 <- 100 / coef(RateCrmodel)[2]
RateCr025 <- coef(RateCrmodel)[2] * x025 #umol mol-1 min-1
RateCbmodel <- lm(Cb025 ~ Counter)
RateCb025 <- coef(RateCbmodel)[2] * x025 #umol mol-1 min-1
RateCimodel <- lm(Ci025 ~ Counter)
RateCi025 <- coef(RateCimodel)[2] * x025 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc025 <- coef(RateCcmodel)[2] * x025 #umol mol-1 min-1
res025 <- 1 / gm025 + 1 / BLC + 1 / gsw

# 另一个正常的叶肉导度
gm0125 <- 0.125 #mol m-2 s-1
Ci0125 <- A / gm0125 + Cc #umol mol-1
Cim0125 <- Ci0125
Cb0125 <- A / gsw + Ci0125 #umol mol-1
Cr0125 <- A / BLC + Cb0125 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr0125 ~ Counter)
```

```
x0125 <- 100 / coef(RateCrmodel)[2]
RateCr0125 <- coef(RateCrmodel)[2] * x0125 #umol mol-1 min-1
RateCbmodel <- lm(Cb0125 ~ Counter)
RateCb0125 <- coef(RateCbmodel)[2] * x0125 #umol mol-1 min-1
RateCimodel <- lm(Ci0125 ~ Counter)
RateCi0125 <- coef(RateCimodel)[2] * x0125 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc0125 <- coef(RateCcmodel)[2] * x0125 #umol mol-1 min-1
res0125 <- 1 / gm0125 + 1 / BLC + 1 / gsw

# 利用不同叶肉导度的数据计算结果构造数据
Ratesgm <-
  c(RateCc0125, RateCc025, RateCc05, RateCc1, RateCc2, RateCc4)
gmval <- c(0.125, 0.25, 0.5, 1, 2, 4)
totalresgm <- c(res0125, res025, res05, res1, res2, res4)
resistance <-
  c(
    rep(res0125, 376),
    rep(res025, 376),
    rep(res05, 376),
    rep(res1, 376),
    rep(res2, 376),
    rep(res4, 376)
  )

# 其余部分与上面类似
# 此时采用不同的气孔导度构建数据
BLC <- 2 #mol m-2 s-1
gm <- 1 #mol m-2 s-1

gsw <- 0.4 #mol m-2 s-1
Ci1 <- A / gm + Cc #umol mol-1
```

```
Cis04 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc04 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res04 <- 1 / gm + 1 / BLC + 1 / gsw

gsw <- 0.2 #mol m-2 s-1
Ci1 <- A / gm + Cc #umol mol-1
Cis02 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc02 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res02 <- 1 / gm + 1 / BLC + 1 / gsw
```

```
gsw <- 0.1 #mol m-2 s-1
Ci1 <- A / gm + Cc #umol mol-1
Cis01 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc01 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res01 <- 1 / gm + 1 / BLC + 1 / gsw

gsw <- 0.05 #mol m-2 s-1
Ci1 <- A / gm + Cc #umol mol-1
Cis05 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc005 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
```

```
res005 <- 1 / gm + 1 / BLC + 1 / gsw

gsw <- 0.025 #mol m-2 s-1
Ci1 <- A / gm + Cc #umol mol-1
Cis0025 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc0025 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res0025 <- 1 / gm + 1 / BLC + 1 / gsw

Ratesgsw <- c(RateCc0025, RateCc005, RateCc01, RateCc02, RateCc04)
gswvals <- c(0.025, 0.05, 0.1, 0.2, 0.4)
totalresgsw <- c(res0025, res005, res01, res02, res04)

# 下面的代码是采用不同的边界层导度
# 含义与上面代码相似
gm <- 1 #mol m-2 s-1
gsw <- 0.4 #mol m-2 s-1

BLC <- 2 #mol m-2 s-1

Ci1 <- A / gm + Cc #umol mol-1
Cib2 <- Ci1
```

```
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc2 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res2 <- 1 / gm + 1 / BLC + 1 / gsw

BLC <- 4 #mol m-2 s-1

Ci1 <- A / gm + Cc #umol mol-1
Cib4 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc4 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res4 <- 1 / gm + 1 / BLC + 1 / gsw
```

```
BLC <- 1 #mol m-2 s-1

Ci1 <- A / gm + Cc #umol mol-1
Cib1 <- Ci1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc1 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res1 <- 1 / gm + 1 / BLC + 1 / gsw

BLC <- 0.5 #mol m-2 s-1

Ci1 <- A / gm + Cc #umol mol-1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cib05 <- Ci1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
```

```
RateCcmodel <- lm(Cc ~ Counter)
RateCc05 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res05 <- 1 / gm + 1 / BLC + 1 / gsw

BLC <- 0.25 #mol m-2 s-1

Ci1 <- A / gm + Cc #umol mol-1
Cb1 <- A / gsw + Ci1 #umol mol-1
Cib025 <- Ci1
Cr1 <- A / BLC + Cb1 #umol mol-1
Counter <- as.numeric(c(1:length(Cc)))
RateCrmodel <- lm(Cr1 ~ Counter)
x1 <- 100 / coef(RateCrmodel)[2]
RateCr1 <- coef(RateCrmodel)[2] * x1 #umol mol-1 min-1
RateCbmodel <- lm(Cb1 ~ Counter)
RateCb1 <- coef(RateCbmodel)[2] * x1 #umol mol-1 min-1
RateCimodel <- lm(Ci1 ~ Counter)
RateCi1 <- coef(RateCimodel)[2] * x1 #umol mol-1 min-1
RateCcmodel <- lm(Cc ~ Counter)
RateCc025 <- coef(RateCcmodel)[2] * x1 #umol mol-1 min-1
res025 <- 1 / gm + 1 / BLC + 1 / gsw

BLCRates <- c(RateCc025, RateCc05, RateCc1, RateCc2, RateCc4)
BLCvals <- c(0.25, 0.5, 1, 2, 4)
totalresBLC <- c(res025, res05, res1, res2, res4)

Scenario <-
  c(
    rep("Boundary Layer Conductance", 5),
    rep("Stomatal Conductance", 5),
    rep("Mesophyll Conductance", 6)
  )
```

```
Rates <- c(BLCRates, Ratesgsw, Ratesgm)
Conductances <- c(BLCvals, gswvals, gmval)
TotalRes <- c(totalresBLC, totalresgsw, totalresgm)

Cidiffusion <- c(Cim0125, Cim025, Cim05, Cim1, Cim2, Cim4)
Adiffusion <- rep(A, 6)
Aappdiffusion <- rep(Aapparent, 6)
variable <- c(rep("Mesophyll Conductance", 6 * length(Cim1)))
conductance <-
  c(rep(0.125, 376),
    rep(0.25, 376),
    rep(0.5, 376),
    rep(1, 376),
    rep(2, 376),
    rep(4, 376))


Diffusionplot <-
  as.data.frame(cbind(
    Cidiffusion,
    Adiffusion,
    Aappdiffusion,
    variable,
    conductance,
    resistance
  ))
write.csv(Diffusionplot, "./data/DiffusionLimitsACI.csv")

Diffusion <-
  as.data.frame(cbind(Scenario, Rates, Conductances, TotalRes))
write.csv(Diffusion, "./data/DiffusionLimits.csv")
```

```
knitr::kable(head(Diffusion))
```

	Scenario	Rates	Conductances	TotalRes
Counter	Boundary Layer Conductance	57.1493939574751	0.25	7.5
Counter.1	Boundary Layer Conductance	64.522239461912	0.5	5.5
Counter.2	Boundary Layer Conductance	68.9712299728077	1	4.5
Counter.3	Boundary Layer Conductance	71.4340185663793	2	4
Counter.4	Boundary Layer Conductance	72.7325667866589	4	3.75
Counter.5	Stomatal Conductance	19.4216526694884	0.025	41.5

10.10.1 扩散限制滞后性

下面的代码，是根据上面代码的计算结果，结合最初的扩散时间的公式，来计算出各个参数的最大最小值，中间值，构造数据：

```
gtot1 <- 1 / res1
t1low = (dZlow) ^ 2 / (gtot1 * dZlow / phat)
t1median = (dZmedian) ^ 2 / (gtot1 * dZmedian / phat)
t1high = (dZhigh) ^ 2 / (gtot1 * dZhigh / phat)
Cc1low <- c(Cc + (t1low * RateCc1))
Cc1median <- c(Cc + (t1median * RateCc1))
Cc1high <- c(Cc + (t1high * RateCc1))
vo1low <- Vomax * O2 / (O2 + Ko * (1 + Cc1low / Kc))
A1low <- vc - 0.5 * vo1low - R
vo1median <- Vomax * O2 / (O2 + Ko * (1 + Cc1median / Kc))
A1median <- vc - 0.5 * vo1median - R
vo1high <- Vomax * O2 / (O2 + Ko * (1 + Cc1high / Kc))
A1high <- vc - 0.5 * vo1high - R

gtot0125 <- 1 / res0125
t0125low = (dZlow) ^ 2 / (gtot0125 * dZlow / phat)
```

```
t0125median = (dZmedian) ^ 2 / (gtot0125 * dZmedian / phat)
t0125high = (dZhigh) ^ 2 / (gtot0125 * dZhigh / phat)
Cc0125low <- c(Cc + (t0125low * RateCc0125))
Cc0125median <- c(Cc + (t0125median * RateCc0125))
Cc0125high <- c(Cc + (t0125high * RateCc0125))
vo0125low <- Vomax * O2 / (O2 + Ko * (1 + Cc0125low / Kc))
A0125low <- vc - 0.5 * vo0125low - R
vo0125median <- Vomax * O2 / (O2 + Ko * (1 + Cc0125median / Kc))
A0125median <- vc - 0.5 * vo0125median - R
vo0125high <- Vomax * O2 / (O2 + Ko * (1 + Cc0125high / Kc))
A0125high <- vc - 0.5 * vo0125high - R

gtot025 <- 1 / res025
t025low = (dZlow) ^ 2 / (gtot025 * dZlow / phat)
t025median = (dZmedian) ^ 2 / (gtot025 * dZmedian / phat)
t025high = (dZhigh) ^ 2 / (gtot025 * dZhigh / phat)
Cc025low <- c(Cc + (t025low * RateCc025))
Cc025median <- c(Cc + (t025median * RateCc025))
Cc025high <- c(Cc + (t025high * RateCc025))
vo025low <- Vomax * O2 / (O2 + Ko * (1 + Cc025low / Kc))
A025low <- vc - 0.5 * vo025low - R
vo025median <- Vomax * O2 / (O2 + Ko * (1 + Cc025median / Kc))
A025median <- vc - 0.5 * vo025median - R
vo025high <- Vomax * O2 / (O2 + Ko * (1 + Cc025high / Kc))
A025high <- vc - 0.5 * vo025high - R

gtot05 <- 1 / res05
t05low = (dZlow) ^ 2 / (gtot05 * dZlow / phat)
t05median = (dZmedian) ^ 2 / (gtot05 * dZmedian / phat)
t05high = (dZhigh) ^ 2 / (gtot05 * dZhigh / phat)
Cc05low <- c(Cc + (t05low * RateCc05))
Cc05median <- c(Cc + (t05median * RateCc05))
```

```
Cc05high <- c(Cc + (t05high * RateCc05))
vo05low <- Vomax * O2 / (O2 + Ko * (1 + Cc05low / Kc))
A05low <- vc - 0.5 * vo05low - R
vo05median <- Vomax * O2 / (O2 + Ko * (1 + Cc05median / Kc))
A05median <- vc - 0.5 * vo05median - R
vo05high <- Vomax * O2 / (O2 + Ko * (1 + Cc05high / Kc))
A05high <- vc - 0.5 * vo05high - R

gtot2 <- 1 / res2
t2low = (dZlow) ^ 2 / (gtot2 * dZlow / phat)
t2median = (dZmedian) ^ 2 / (gtot2 * dZmedian / phat)
t2high = (dZhigh) ^ 2 / (gtot2 * dZhigh / phat)
Cc2low <- c(Cc + (t2low * RateCc2))
Cc2median <- c(Cc + (t2median * RateCc2))
Cc2high <- c(Cc + (t2high * RateCc2))
vo2low <- Vomax * O2 / (O2 + Ko * (1 + Cc2low / Kc))
A2low <- vc - 0.5 * vo2low - R
vo2median <- Vomax * O2 / (O2 + Ko * (1 + Cc2median / Kc))
A2median <- vc - 0.5 * vo2median - R
vo2high <- Vomax * O2 / (O2 + Ko * (1 + Cc2high / Kc))
A2high <- vc - 0.5 * vo2high - R

gtot4 <- 1 / res4
t4low = (dZlow) ^ 2 / (gtot4 * dZlow / phat)
t4median = (dZmedian) ^ 2 / (gtot4 * dZmedian / phat)
t4high = (dZhigh) ^ 2 / (gtot4 * dZhigh / phat)
Cc4low <- c(Cc + (t4low * RateCc4))
Cc4median <- c(Cc + (t4median * RateCc4))
Cc4high <- c(Cc + (t4high * RateCc4))
vo4low <- Vomax * O2 / (O2 + Ko * (1 + Cc4low / Kc))
A4low <- vc - 0.5 * vo4low - R
vo4median <- Vomax * O2 / (O2 + Ko * (1 + Cc4median / Kc))
```

```
A4median <- vc - 0.5 * vo4median - R
vo4high <- Vomax * O2 / (O2 + Ko * (1 + Cc4high / Kc))
A4high <- vc - 0.5 * vo4high - R

Cc0125high2 <- c(Cc + (t0125high * RateCc0125 * 2))
Cc025high2 <- c(Cc + (t025high * RateCc025 * 2))
Cc05high2 <- c(Cc + (t05high * RateCc05 * 2))
Cc1high2 <- c(Cc + (t1high * RateCc1 * 2))
Cc2high2 <- c(Cc + (t2high * RateCc2 * 2))
Cc4high2 <- c(Cc + (t4high * RateCc4 * 2))

vo0125high2 <- Vomax * O2 / (O2 + Ko * (1 + Cc0125high2 / Kc))
A0125high2 <- vc - 0.5 * vo0125high2 - R
vo025high2 <- Vomax * O2 / (O2 + Ko * (1 + Cc025high2 / Kc))
A025high2 <- vc - 0.5 * vo025high2 - R
vo05high2 <- Vomax * O2 / (O2 + Ko * (1 + Cc05high2 / Kc))
A05high2 <- vc - 0.5 * vo05high2 - R
vo1high2 <- Vomax * O2 / (O2 + Ko * (1 + Cc1high2 / Kc))
A1high2 <- vc - 0.5 * vo1high2 - R
vo2high2 <- Vomax * O2 / (O2 + Ko * (1 + Cc2high2 / Kc))
A2high2 <- vc - 0.5 * vo2high2 - R
vo4high2 <- Vomax * O2 / (O2 + Ko * (1 + Cc4high2 / Kc))
A4high2 <- vc - 0.5 * vo4high2 - R

Ahigh2 <-
  c(A0125high2, A025high2, A05high2, A1high2, A2high2, A4high2)

Cc0125high3 <- c(Cc + (t0125high * RateCc0125 * 3))
Cc025high3 <- c(Cc + (t025high * RateCc025 * 3))
Cc05high3 <- c(Cc + (t05high * RateCc05 * 3))
Cc1high3 <- c(Cc + (t1high * RateCc1 * 3))
Cc2high3 <- c(Cc + (t2high * RateCc2 * 3))
```

```
Cc4high3 <- c(Cc + (t4high * RateCc4 * 3))

vo0125high3 <- Vomax * O2 / (O2 + Ko * (1 + Cc0125high3 / Kc))
A0125high3 <- vc - 0.5 * vo0125high3 - R
vo025high3 <- Vomax * O2 / (O2 + Ko * (1 + Cc025high3 / Kc))
A025high3 <- vc - 0.5 * vo025high3 - R
vo05high3 <- Vomax * O2 / (O2 + Ko * (1 + Cc05high3 / Kc))
A05high3 <- vc - 0.5 * vo05high3 - R
vo1high3 <- Vomax * O2 / (O2 + Ko * (1 + Cc1high3 / Kc))
A1high3 <- vc - 0.5 * vo1high3 - R
vo2high3 <- Vomax * O2 / (O2 + Ko * (1 + Cc2high3 / Kc))
A2high3 <- vc - 0.5 * vo2high3 - R
vo4high3 <- Vomax * O2 / (O2 + Ko * (1 + Cc4high3 / Kc))
A4high3 <- vc - 0.5 * vo4high3 - R

Ahigh3 <-
  c(A0125high3, A025high3, A05high3, A1high3, A2high3, A4high3)

Cidiffusion <- c(Cim0125, Cim025, Cim05, Cim1, Cim2, Cim4)
Alow <- c(A0125low, A025low, A05low, A1low, A2low, A4low)
Amedian <-
  c(A0125median,
    A025median,
    A05median,
    A1median,
    A2median,
    A4median)
Ahigh <- c(A0125high, A025high, A05high, A1high, A2high, A4high)

variable <- c(rep("Mesophyll Conductance", 6 * length(Cim1)))
conductance <-
  c(rep(0.125, 376),
```

```
    rep(0.25, 376),
    rep(0.5, 376),
    rep(1, 376),
    rep(2, 376),
    rep(4, 376))

Diffusionplot2 <-
  as.data.frame(
    cbind(
      Cidiffusion,
      Alow,
      Amedian,
      Ahigh,
      Ahigh2,
      Ahigh3,
      variable,
      conductance,
      resistance
    )
  )
write.csv(Diffusionplot2, "./data/DiffusionLimitsACI2.csv")
```

最终够到的不同导度下的扩散数据如下：

```
knitr::kable(head(Diffusionplot2))
```

Cidiffusion	Alow	Amedian	Ahigh	Ahigh2
-14.0805894452511	-4.87461753940542	-4.86419435199041	-4.81536251060599	-4.74710314038274
-11.8541114570766	-4.72133615031493	-4.71094118846005	-4.66224099010836	-4.59416400241068
-9.63096032171488	-4.56847050260645	-4.55810365181624	-4.50953456487788	-4.44163922971142
-7.41112252124025	-4.41601890716257	-4.40568005356001	-4.35724154927421	-4.28952714053524
-5.19458461086401	-4.26397968400382	-4.25366871432613	-4.20536026677158	-4.13782606222113
-2.98133321844053	-4.11235116222692	-4.10206796382144	-4.05388904989726	-3.98653433113539

10.11 扩散限制作图

```
data <- read.csv("./data/DiffusionLimitsACI2.csv")
data$resistance <- as.factor(data$resistance)
graph <- ggplot(data, aes(x = Cidiffusion, y = Ahigh3, colour = resistance)) +
  geom_abline(slope=0,intercept=0,size=1.5)+
  geom_point()+
  labs(colour = 'Total Resistance')+
  scale_colour_brewer(palette = 'Spectral') +
  theme_bw() +
  scale_x_continuous(limits=c(0,100))+
  scale_y_continuous(limits=c(-5,10))+
  theme(
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 15),
    legend.position = 'right',
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank()
  )
graph
```

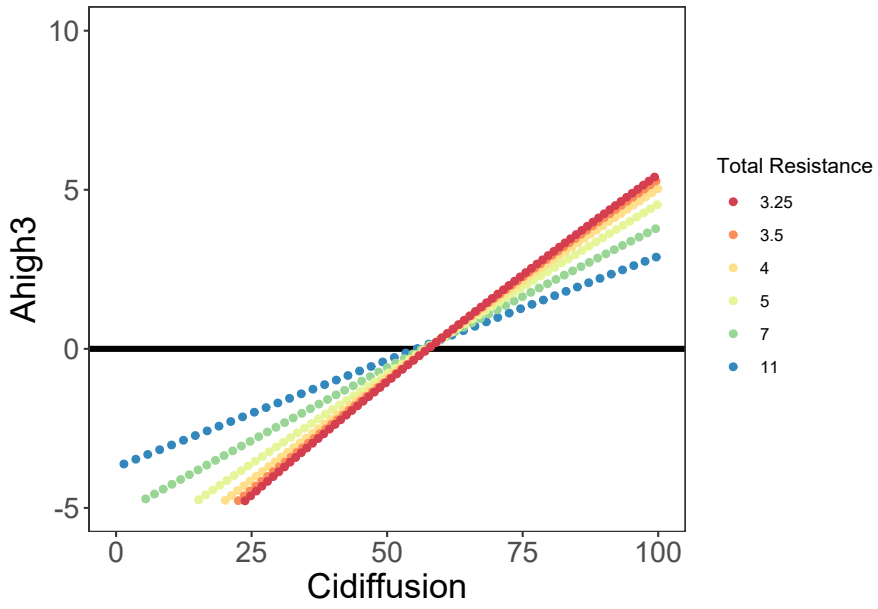


图 10.18: 不同 Ci 扩散限制下的差异

```
datahigh <- data[data$resistance == "11",]
graph <- ggplot(datahigh, aes(x = Cidiffusion, y = Ahigh3)) +
  geom_abline(slope=0,intercept=0,size=1.5)+
  geom_point(colour = "red")+
  geom_point(aes(x = Cidiffusion, y = Ahigh2), colour = "blue")+
  geom_point(aes(x = Cidiffusion, y = Ahigh), colour = "green")+
  labs(colour = 'Total Resistance')+
  scale_colour_brewer(palette = 'Spectral') +
  theme_bw() +
  scale_x_continuous(limits=c(0,100))+
  scale_y_continuous(limits=c(-2.5,2.5))+
  theme(
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 15),
    legend.position = 'right',
```

```
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank()
  )
graph
```

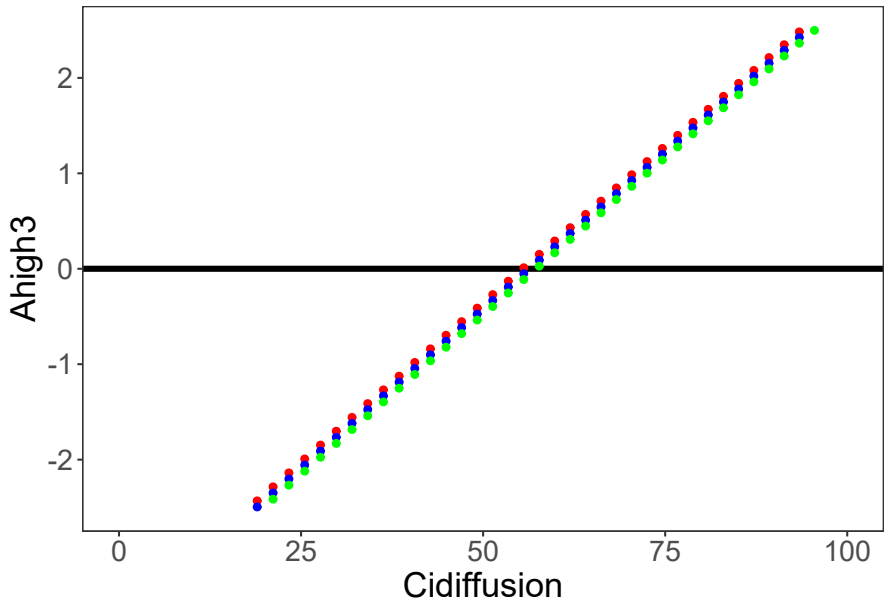


图 10.19: 不同 Ci 扩散限制下的差异（resistance = 11）

```
data2 <- read.csv("./data/DiffusionLimits.csv")
graph <- ggplot(data2, aes(x = TotalRes, y = Rates, colour = Scenario)) +
  geom_point()+
  scale_colour_brewer(palette = 'Spectral') +
  theme_bw() +
  theme(
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 12),
    panel.grid.major = element_blank(),
```

```
    panel.grid.minor = element_blank()
  )
graph
```

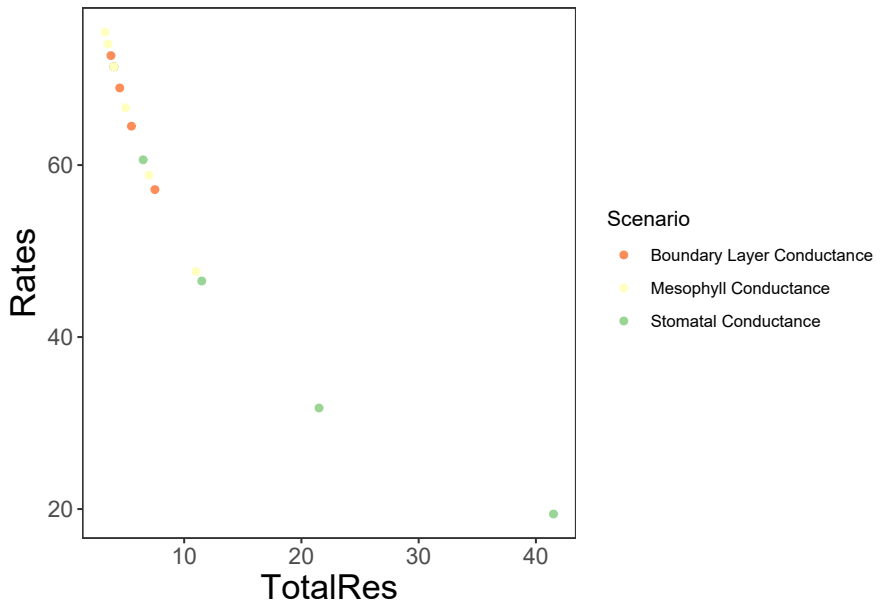


图 10.20: 不同总导度下的各个导度的速率变化

```
m1 <- lm(Rates ~ I(1/TotalRes)+I((1/TotalRes)^2),data=data2)
summary(m1)
```

```
##
## Call:
## lm(formula = Rates ~ I(1/TotalRes) + I((1/TotalRes)^2), data = data2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3590 -0.9844 -0.1236  0.7706  2.4510
```

```
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         13.042      1.684   7.745 3.18e-06 ***
## I(1/TotalRes)      421.928     22.487  18.763 8.49e-11 ***
## I((1/TotalRes)^2) -737.690     65.500 -11.262 4.46e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.705 on 13 degrees of freedom
## Multiple R-squared:  0.9904, Adjusted R-squared:  0.9889
## F-statistic: 668.8 on 2 and 13 DF,  p-value: 7.799e-14
```

```
ab <- predict(m1)
red <- 1/data2$TotalRes
plot(ab~red)
```

10.11.1 补偿点的计算

计算补偿点，代码同前文类似，只是采用了不同导度下的数值：

```
data <- read.csv("./data/DiffusionLimitsACI2.csv")
#Gamma
# For Ci-based estimates, only use Ci < 100
dataCi <- data[data$Cidiffusion < 100, ]
dataCi0125 <- dataCi[dataCi$conductance == "0.125", ]
dataCi025 <- dataCi[dataCi$conductance == "0.25", ]
dataCi05 <- dataCi[dataCi$conductance == "0.5", ]
dataCi1 <- dataCi[dataCi$conductance == "1", ]
dataCi2 <- dataCi[dataCi$conductance == "2", ]
```

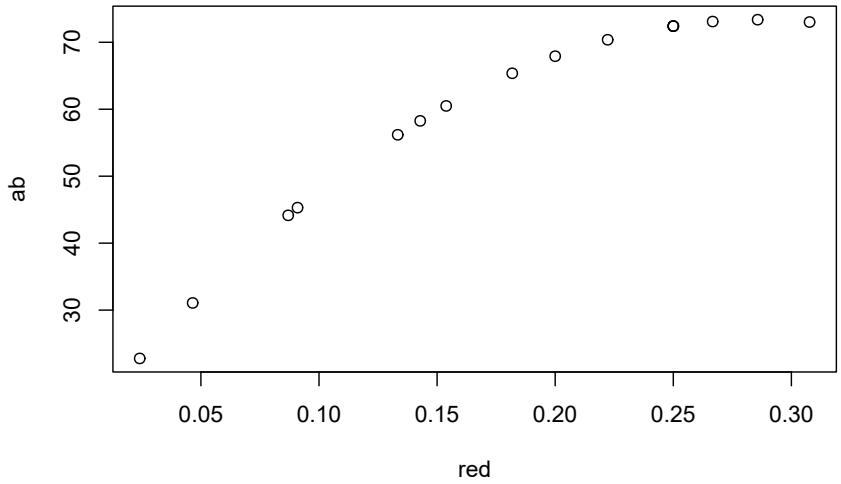


图 10.21: 不同阻力下的各个导度的速率变化预测值

```
dataCi4 <- dataCi[dataCi$conductance == "4", ]
m1 <- lm(dataCi0125$Ahigh ~ dataCi0125$Cidiffusion)
summary(m1)
```

```
##
## Call:
## lm(formula = dataCi0125$Ahigh ~ dataCi0125$Cidiffusion)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.040649 -0.014495  0.005307  0.017054  0.021232
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -3.836e+00  4.318e-03  -888.5   <2e-16 ***
```

```
## dataCi0125$Cidiffusion  6.665e-02  7.873e-05   846.6   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01934 on 52 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 7.167e+05 on 1 and 52 DF,  p-value: < 2.2e-16
```

```
Gamma0125 <- -m1$coefficients[1] / m1$coefficients[2]
m2 <- lm(dataCi025$Ahigh ~ dataCi025$Cidiffusion)
summary(m2)
```

```
##
## Call:
## lm(formula = dataCi025$Ahigh ~ dataCi025$Cidiffusion)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.070876 -0.025427  0.009071  0.029765  0.036593
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -5.2507346  0.0092187  -569.6   <2e-16 ***
## dataCi025$Cidiffusion  0.0904227  0.0001544   585.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03329 on 59 degrees of freedom
## Multiple R-squared:  0.9998, Adjusted R-squared:  0.9998
## F-statistic: 3.428e+05 on 1 and 59 DF,  p-value: < 2.2e-16
```

```
Gamma025 <- -m2$coefficients[1] / m2$coefficients[2]
m3 <- lm(dataCi05$Ahigh ~ dataCi05$Cidiffusion)
summary(m3)
```

```
##
## Call:
## lm(formula = dataCi05$Ahigh ~ dataCi05$Cidiffusion)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.10435 -0.03583  0.01358  0.04311  0.05340
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -6.4051036  0.0150363    -426   <2e-16 ***
## dataCi05$Cidiffusion  0.1097686  0.0002391     459   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04851 on 65 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 2.107e+05 on 1 and 65 DF,  p-value: < 2.2e-16
```

```
Gamma05 <- -m3$coefficients[1] / m3$coefficients[2]
m4 <- lm(dataCi1$Ahigh ~ dataCi1$Cidiffusion)
summary(m4)
```

```
##
## Call:
## lm(formula = dataCi1$Ahigh ~ dataCi1$Cidiffusion)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.13150 -0.04513  0.01709  0.05402  0.06692
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -7.1817472  0.0199135  -360.6   <2e-16 ***
## dataCi1$Cidiffusion  0.1227771  0.0003085   398.0   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06074 on 69 degrees of freedom
## Multiple R-squared:  0.9996, Adjusted R-squared:  0.9996
## F-statistic: 1.584e+05 on 1 and 69 DF,  p-value: < 2.2e-16
```

```
Gamma1 <- -m4$coefficients[1] / m4$coefficients[2]
m5 <- lm(dataCi2$Ahigh ~ dataCi2$Cidiffusion)
summary(m5)
```

```
##
## Call:
## lm(formula = dataCi2$Ahigh ~ dataCi2$Cidiffusion)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14803 -0.04874  0.01837  0.05995  0.07511
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -7.6427727  0.0230595  -331.4   <2e-16 ***
## dataCi2$Cidiffusion  0.1305083  0.0003538   368.9   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06815 on 71 degrees of freedom
## Multiple R-squared:  0.9995, Adjusted R-squared:  0.9995
## F-statistic: 1.361e+05 on 1 and 71 DF,  p-value: < 2.2e-16
```

```
Gamma2 <- -m5$coefficients[1] / m5$coefficients[2]
m6 <- lm(dataCi4$Ahigh ~ dataCi4$Cidiffusion)
summary(m6)
```

```
##
## Call:
## lm(formula = dataCi4$Ahigh ~ dataCi4$Cidiffusion)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.15721 -0.05258  0.01994  0.06470  0.07968
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -7.8955772  0.0248634  -317.6   <2e-16 ***
## dataCi4$Cidiffusion  0.1347503  0.0003799   354.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.07225 on 72 degrees of freedom
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9994
## F-statistic: 1.258e+05 on 1 and 72 DF,  p-value: < 2.2e-16
```

```
Gamma4 <- -m6$coefficients[1] / m6$coefficients[2]

GammaCi <- c(Gamma0125, Gamma025, Gamma05, Gamma1, Gamma2, Gamma4)
```

10.11.2 所有图形代码

```
data <- read.csv("./data/PRdata.csv")
data$Ccfull <- as.numeric(data$Ccfull)
data$Delay <- as.factor(data$Delay)
cols <- gg_color_hue(6)
Panel_1A <-
  ggplot(data, aes(
    x = Cifull,
    y = Anet,
    colour = Delay,
    linetype = Delay
  )) +
  # 零水平参考线
  geom_abline(slope = 0,
              intercept = 0,
              size = 1.5) +
  #ci 与 A loess 方法的拟合曲线
  geom_smooth(method = "loess", se = FALSE, size = 2) +
  scale_linetype_manual(
    name = "Delay (s)",
    labels = c("0", "15", "30", "60", "120", "300"),
    values = c("solid", "longdash", "twodash", "dotdash", "dashed", "dotted")
  ) +
  ggtitle(expression(paste(
    bold("(a)"), " Modelled Photorespiratory Effect"
```

```
))) +
labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
  -1
} * ")"),
y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
  -2
} ~ s ^ {
  -1
} * ")")) +
labs(colour = 'Delay (s)') +
scale_x_continuous(limits = c(0, 250)) +
scale_y_continuous(limits = c(-5, 20)) +
scale_colour_manual(
  values = cols,
  name = "Delay (s)",
  labels = c("0", "15", "30", "60", "120", "300")
) +
theme_bw() +
theme(
  axis.title.x = element_text(size = 20),
  axis.title.y = element_text(size = 20),
  axis.line = element_line(size = 1),
  axis.ticks = element_line(size = 1.5),
  axis.ticks.length = unit(1.5, "mm"),
  rect = element_rect(size = 2),
  axis.text.x = element_text(size = 14, color = 'black'),
  axis.text.y =
    element_text(
      size = 14,
      color = 'black',
      hjust = (1)
    ),
```

```
    legend.position = c(0.15, 0.75),
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 12, color = "black"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 14)
  )
Panel_1A
```

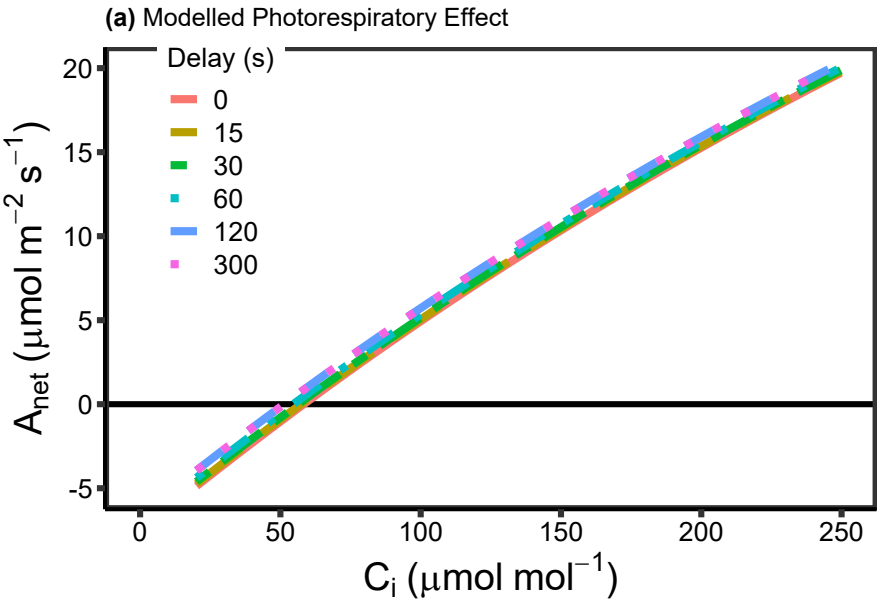


图 10.22: 不同时间滞后性的 Anet VS Ci

```
Panel_1A_inset <-
  ggplot(data, aes(
    x = Cifull,
    y = Anet,
```

```
    colour = Delay,
    linetype = Delay
  )) +
  geom_abline(slope = 0,
              intercept = 0,
              size = 1.5) +
  geom_smooth(method = "loess", se = FALSE, size = 2) +
  scale_linetype_manual(
    name = "Delay (s)",
    labels = c("0", "15", "30", "60", "120", "300"),
    values = c("solid", "longdash", "twodash", "dotdash", "dashed", "dotted")
  ) +
  labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")"),
  y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")")) +
  labs(colour = 'Delay (s)') +
  scale_x_continuous(limits = c(45, 65)) +
  scale_y_continuous(limits = c(-1, 1), breaks = c(-1, 0, 1)) +
  scale_colour_manual(values = cols) +
  theme_bw() +
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    axis.line = element_line(size = 1),
    axis.ticks = element_line(size = 1.5),
    axis.ticks.length = unit(1.5, "mm"),
    rect = element_rect(size = 2),
```

```
    axis.text.x = element_text(size = 14, color = 'black'),
    axis.text.y =
      element_text(
        size = 14,
        color = 'black',
        hjust = (1)
      ),
    axis.title = element_blank(),
    axis.text = element_text(size = 12, color = "black"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    legend.position = 'none',
    plot.background = element_blank()
  )



cols <- gg_color_hue(6)

data <- read.csv("./data/DiffusionLimitsACI2.csv")
data$resistance <- as.factor(data$resistance)
Panel_1Db <-
  ggplot(data,
         aes(
           x = Cidiffusion,
           y = Ahigh,
           colour = resistance,
           linetype = resistance
         )) +
  geom_abline(slope = 0,
              intercept = 0,
              size = 1.5) +
```

```
  geom_smooth(method = "loess", se = FALSE, size = 2) +
  scale_linetype_manual(
    name = "Total Resistance",
    labels = c("3.25", "3.5", "4", "5", "7", "11"),
    values = c("solid", "longdash", "twodash", "dotdash", "dashed", "dotted")
  ) +
  ggtitle(expression(paste(bold("(d)"), " Modelled Resistance Effect"))) +
  scale_color_manual(
    values = cols,
    name = "Total Resistance",
    labels = c("3.25", "3.5", "4", "5", "7", "11")
  ) +
  theme_bw() +
  labs(x = expression(paste(C[i] ~ "(" * mu * mol ~ mol ^ {
    -1
  } * ")")), y = expression(paste(A[net] ~ "(" * mu * mol ~ m ^ {
    -2
  } ~ s ^ {
    -1
  } * ")"))) +
  scale_x_continuous(limits = c(0, 250)) +
  scale_y_continuous(limits = c(-5, 20)) +
  theme(
    axis.title.x = element_text(size = 20),
    axis.title.y = element_text(size = 20),
    axis.line = element_line(size = 1),
    axis.ticks = element_line(size = 1.5),
    axis.ticks.length = unit(1.5, "mm"),
    rect = element_rect(size = 2),
    axis.text.x = element_text(size = 14, color = 'black'),
    axis.text.y =
      element_text(
```

```
        size = 14,
        color = 'black',
        hjust = (1)
      ),
    legend.position = c(0.22, 0.75),
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 12, color = "black"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 14)
  )
Panel_1Db
```

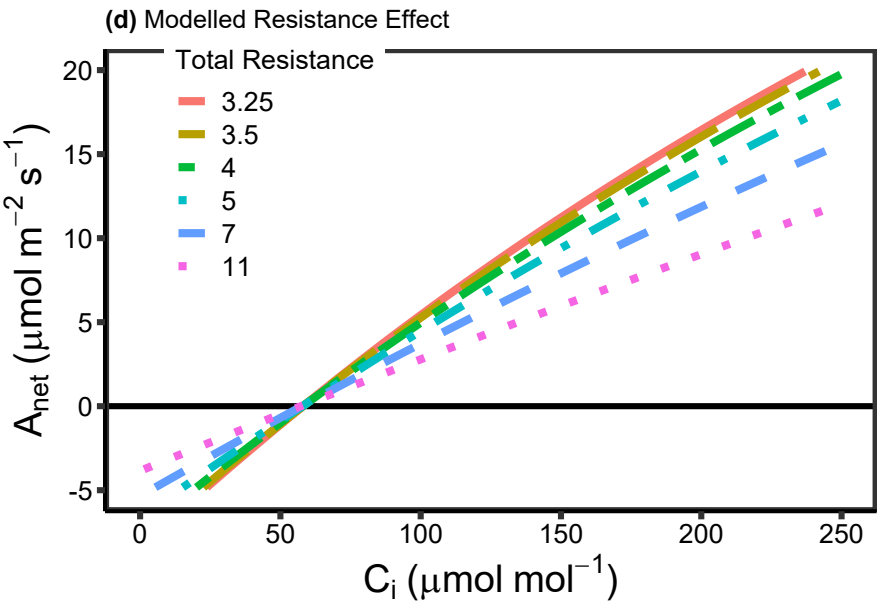


图 **10.23:** 不同总阻力下的 Anet VS Ci

```
data <- read.csv("./data/DiffusionLimitsACI2.csv")
data$resistance <- as.factor(data$resistance)
data <- data[data$resistance == "11",]
resistance <- rep(data$resistance, 3)
Cidiffusion <- rep(data$Cidffusion, 3)
A <- c(data$Ahigh, data$Ahigh2, data$Ahigh3)
Rate <-
  c(rep(100, length(data$Ahigh)), rep(200, length(data$Ahigh2)), rep(300, length(data
data3 <- rbind(data, data, data)
data3$A <- c(data$Ahigh, data$Ahigh2, data$Ahigh3)
data3$Rate <-
  as.factor(c(rep(100, length(data$Ahigh)), rep(200, length(data$Ahigh2)), rep(300, l
Panel_1Dbinset <-
  ggplot(data3, aes(
    x = Cidiffusion,
    y = A,
    colour = Rate,
    linetype = Rate
  )) +
  geom_abline(slope = 0,
              intercept = 0,
              size = 1.5) +
  geom_smooth(method = "loess", se = FALSE, size = 2) +
  scale_linetype_manual(
    name = "Rate",
    labels = c("100", "200", "300"),
    values = c("solid", "longdash", "twodash")
  ) +
  scale_color_manual(
    values = cols,
    name = "Rate",
    labels = c("100", "200", "300")
```

```
) +
theme_bw() +
labs(x = expression(paste(C[i] ~ "(" * mu * mol ~ mol ^ {
  -1
} * ")"), y = expression(paste(A[net] ~ "(" * mu * mol ~ m ^ {
  -2
} ~ s ^ {
  -1
} * ")"))) +
scale_x_continuous(limits = c(45, 65)) +
scale_y_continuous(limits = c(-1, 1), breaks = c(-1, 0, 1)) +
theme(
  axis.title.x = element_blank(),
  axis.title.y = element_blank(),
  axis.line = element_line(size = 1),
  axis.ticks = element_line(size = 1.5),
  axis.ticks.length = unit(1.5, "mm"),
  rect = element_rect(size = 2),
  axis.text.x = element_text(size = 14, color = 'black'),
  axis.text.y =
    element_text(
      size = 14,
      color = 'black',
      hjust = (1)
    ),
  legend.position = c(0.22, 0.75),
  axis.title = element_text(size = 18),
  axis.text = element_text(size = 12, color = "black"),
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank(),
  plot.background = element_blank(),
  legend.text = element_text(size = 12),
```

```
    legend.key = element_blank(),
    legend.title = element_blank(),
    legend.background = element_blank()
  )
Panel_1Dbinset
```

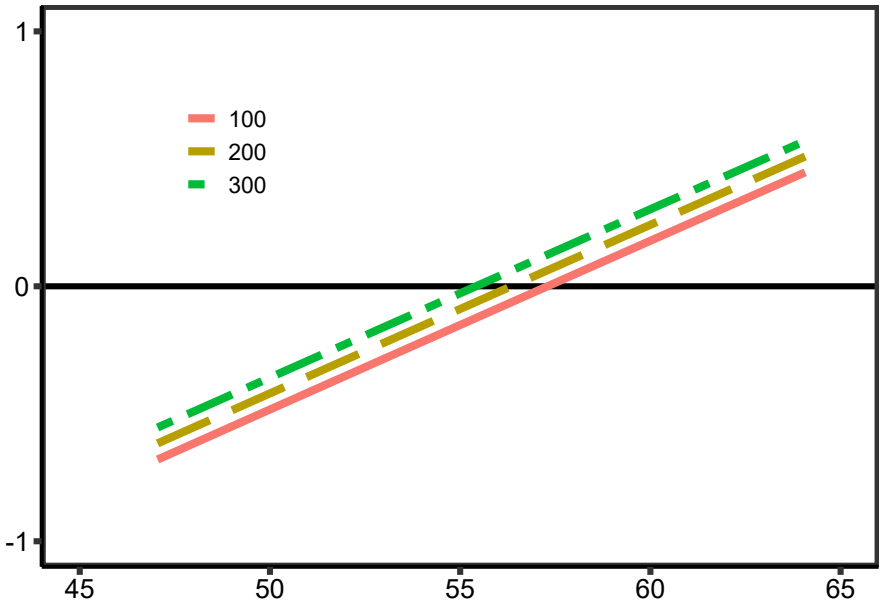


图 **10.24:** 不同导度下的 A VS Ci

```
data2 <- read.csv("./data/DiffusionLimits.csv")
data2$Conductance <-
  revalue(
    data2$Scenario,
    c(
      "Boundary Layer Conductance" = "Boundary Layer",
      "Mesophyll Conductance" = "Mesophyll",
      "Stomatal Conductance" = "Stomatal"
```

```
    )
  )
cols <- gg_color_hue(3)

Panel_1E <-
  ggplot(data2, aes(x = TotalRes, y = Rates, colour = Conductance)) +
  geom_point(size = 4) +
  ggtitle(expression(paste(bold("(e)"), " Modelled Resistance Effect"))) +
  labs(x = expression(paste(r[total] * " (s" ~ m ^ {
    2
  } ~ mol ^ {
    -1
  } * ")")),
  y = expression(paste(C[c] * " Ramp Rate (" * mu * mol ~ mol ^
                                      {
                                        -1
                                      } ~ min ^ {
                                        -1
                                      } * ")"))) +
  scale_x_continuous(limits = c(0, 50)) +
  scale_y_continuous(limits = c(0, 80)) +
  scale_color_manual(
    values = cols,
    name = "Manipulated Resistance",
    labels = c("Boundary Layer", "Mesophyll", "Stomatal")
  ) +
  theme_bw() +
  theme(
    axis.title.x = element_text(size = 20),
    axis.title.y = element_text(size = 20),
    axis.line = element_line(size = 1),
    axis.ticks = element_line(size = 1.5),
```

```
    axis.ticks.length = unit(1.5, "mm"),
    rect = element_rect(size = 2),
    axis.text.x = element_text(size = 14, color = 'black'),
    axis.text.y =
      element_text(
        size = 14,
        color = 'black',
        hjust = (1)
      ),
    legend.position = c(0.70, 0.84),
    axis.title = element_text(size = 18),
    axis.text = element_text(size = 12, color = "black"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    legend.text = element_text(size = 14),
    legend.title = element_text(size = 14)
  )
Panel_1E
```

```
data <- read.csv("./data/RASdata.csv")
data$Ccfull <- as.numeric(data$Ccfull)
data$Deactivation <- as.factor(data$Deactivation)
print(levels(data$Deactivation))
```

```
## [1] "High"   "Low"    "Medium" "None"
```

```
data$Deactivation <-
  factor(data$Deactivation, levels(data$Deactivation)[c(4, 2, 3, 1)])
FigS1 <-
```

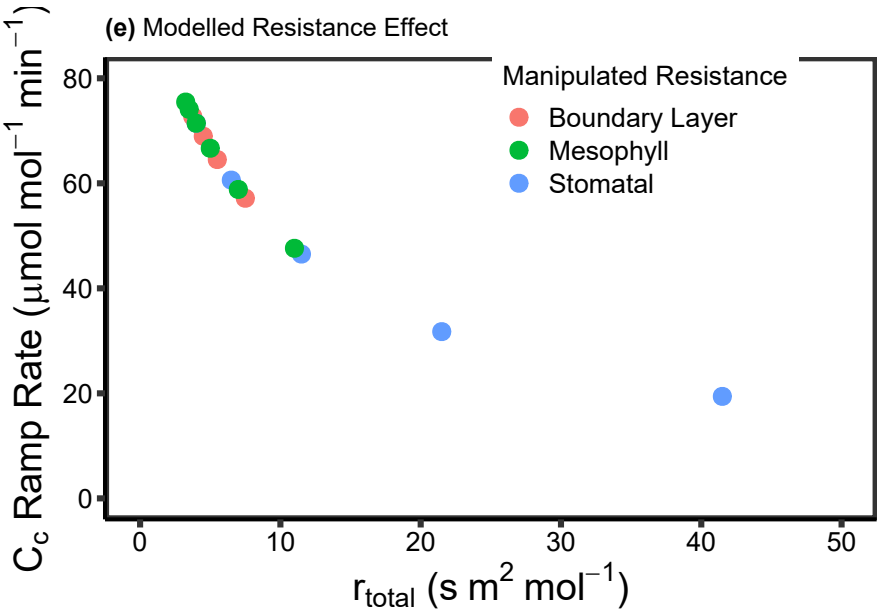


图 **10.25:** 不同总导度下的各个导度的变化

```
ggplot(data,
       aes(
         x = Cifull,
         y = Anet,
         colour = Deactivation,
         linetype = Deactivation
       )) +
geom_smooth(method = "loess", se = FALSE, size = 2) +
scale_linetype_manual(
  name = "Deactivation",
  labels = c("None", "Low", "Medium", "High"),
  values = c("solid", "longdash", "dashed", "dotted")
) +
labs(x = expression(C[i] ~ "(" * mu * mol ~ mol ^ {
  -1
```

```
} * ")"),
y = expression(A[net] ~ "(" * mu * mol ~ m ^ {
  -2
} ~ s ^ {
  -1
} * ")")) +
labs(colour = 'Deactivation') +
scale_x_continuous(limits = c(25, 100),
                   breaks = c(25, 40, 55, 70, 85, 100)) +
scale_y_continuous(limits = c(-5, 5)) +
geom_hline(yintercept = 0, size = 1.5) +
theme_bw() +
theme(
  axis.title.x = element_text(size = 20),
  axis.title.y = element_text(size = 20),
  axis.line = element_line(size = 1),
  axis.ticks = element_line(size = 1.5),
  axis.ticks.length = unit(1.5, "mm"),
  rect = element_rect(size = 2),
  axis.text.x = element_text(size = 14, color = 'black'),
  axis.text.y =
    element_text(
      size = 14,
      color = 'black',
      hjust = (1)
    ),
  legend.position = c(0.8, 0.2),
  legend.text = element_text(size = 16),
  legend.title = element_text(size = 16),
  panel.grid.major = element_blank(),
  panel.grid.minor = element_blank()
```

```
  )
FigS1
```

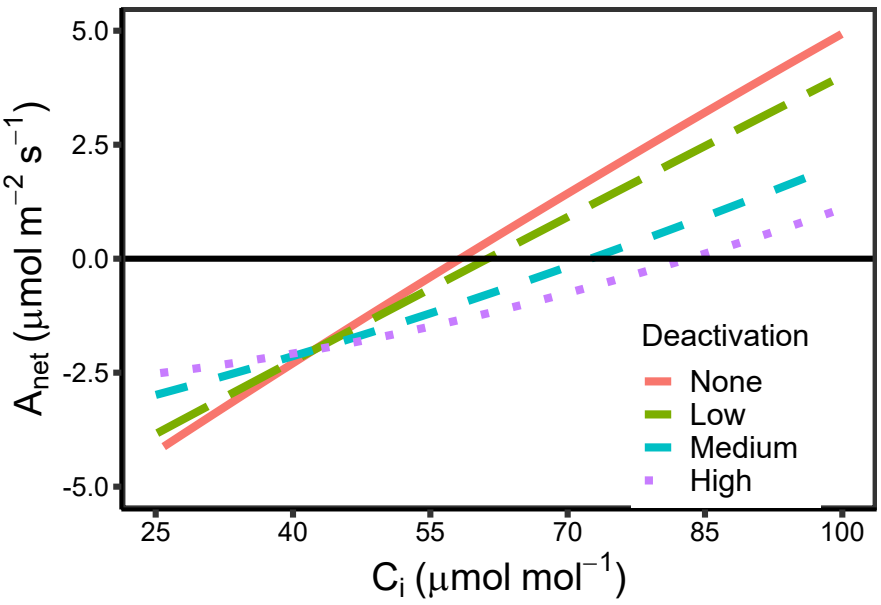


图 10.26: Rubisco 不同失活程度下的下 A VS Ci

10.12 LI-6800 荧光数据分析

LI-6800 能够直接提供基本的叶绿素荧光参数，其他参数均可通过这些基本的参数进行计算，计算也较简单，在此不赘述，需要注意的是快相荧光部分的数据，因为分析 ojip 数据的模型有很多，很多都需要复杂的计算，在此我们先将其中较为简单的 jip test 数据分析进行介绍。

10.12.1 jip test 的实现

LI-6800 增加了 ojip 曲线测量功能，本功能主要是针对测量数据的 jip test 的实现。

10.12.2 `jiptest` 软件包安装

目前 `jiptest` 暂时放在 github 我的软件仓库内，并没提交 CRAN，因此需要 `devtools` 的支持，然后才能从 github 安装 `jiptest`。

```
install.packages("devtools")
library(devtools)
install_github("zhujiedong/jiptest")
```

10.12.3 `read_files` 及 `read_dcfiles` 函数

`read_files` 用于批量读取所有调制光测量数据，方便用于其他的数据分析。**函数要求所有数据必须是 `xlsx` 格式，并且所有测量数据都保存在同一文件夹内。**，如有其他文件或测量数据则会报错。`read_dcfiles` 用于批量读取所有连续光测量数据，其他与 `read_dcfiles` 相同。

函数仅有一个参数，即保存数据文件夹的路径，使用如下：

表 10.2: jiptest 批量导入数据后的样式

SECS	FLUOR	SOURCE
2.41e-05	887	INDUCTION-484-20171225-13_15_58
2.81e-05	998	INDUCTION-484-20171225-13_15_58
3.19e-05	1013	INDUCTION-484-20171225-13_15_58
3.60e-05	1055	INDUCTION-484-20171225-13_15_58
4.01e-05	1044	INDUCTION-484-20171225-13_15_58
4.41e-05	1205	INDUCTION-484-20171225-13_15_58

表 10.3: jiptest DC 数据批量导入数据后的样式

SECS	FLUOR	SOURCE
2.41e-05	154000	INDUCTION-484-20171225-13_15_58
2.81e-05	157566	INDUCTION-484-20171225-13_15_58
3.19e-05	161261	INDUCTION-484-20171225-13_15_58
3.60e-05	165114	INDUCTION-484-20171225-13_15_58
4.01e-05	169069	INDUCTION-484-20171225-13_15_58
4.41e-05	173231	INDUCTION-484-20171225-13_15_58

```
library(jiptest)
jipdata <- read_files("./data/ojip")
```

```
dcjipdata <- read_dcfiles("./data/ojip")
```

10.12.4 jip_test 及 jip_dctest 函数

jiptest 是 jiptest 的核心函数，用于所有数据的 jip test 分析，函数仅包一个参数，测量数据的保存文件夹路径。jip_dctest 与 jip_test 相似，用于连续光测量数据的分析。

```
jip_results <- jip_test("./data/ojip")
```

```
## for the current version, we suggest you use the default settings of
##
##   the duration (1000 ms or more) in the induction settings of LI-6800

## New names:
## * `` -> ...1
## * `` -> ...2
## New names:
## * `` -> ...1
## * `` -> ...2
## New names:
## * `` -> ...1
## * `` -> ...2
## New names:
## * `` -> ...1
## * `` -> ...2
```

表 10.4: jiptest 输出的计算参数

parameters	data_file1	data_file2	data_file3	data_file4
Fo	887.000	849.000	981.000	914.000
Fm	4649.520	3714.270	4225.240	3754.270
F300	2181.400	1798.600	2067.800	2537.200
FJ	2594.300	2104.600	2456.800	2569.000
FI	4443.110	3547.110	4009.220	3556.530

Tfmax	256.016	316.016	272.016	200.016

```
dcjip_results <- jip_dctest("./data/ojip")
```

表 10.5: jiptest DC 数据输出的计算参数

parameters	data_file1	data_file2	data_file3	data_file4
Fo	154000.000	133191.000	156664.000	153589.000
Fm	713839.000	560610.000	647660.000	584839.000
F300	338091.000	275476.800	322859.200	388110.400
FJ	400744.000	320245.000	376885.000	406671.000
FI	681505.000	534314.000	614590.000	552253.000
Tfmax	276.016	300.016	288.016	212.016

计算参数见表 10.4 及 10.5，考虑到排版，仅显示部分内容。若需要将数据导出，可以使用相关命令，如：

```
# export the results of jiptest to a csv file
write.csv(jip_results, "d:/data/jip_resluts.csv")
write.csv(dcjip_results, "d:/data/dcjip_resluts.csv")
```

10.12.5 jip_plot 及 jip_dcplot 函数

jip_plot 和 jip_dcplot 基于 ggplot2，用于快速预览所有测量结果的 ojip 曲线。函数仅有一个参数，即保存数据文件夹的路径，使用如下：

```
jip_plot("./data/ojip")
```

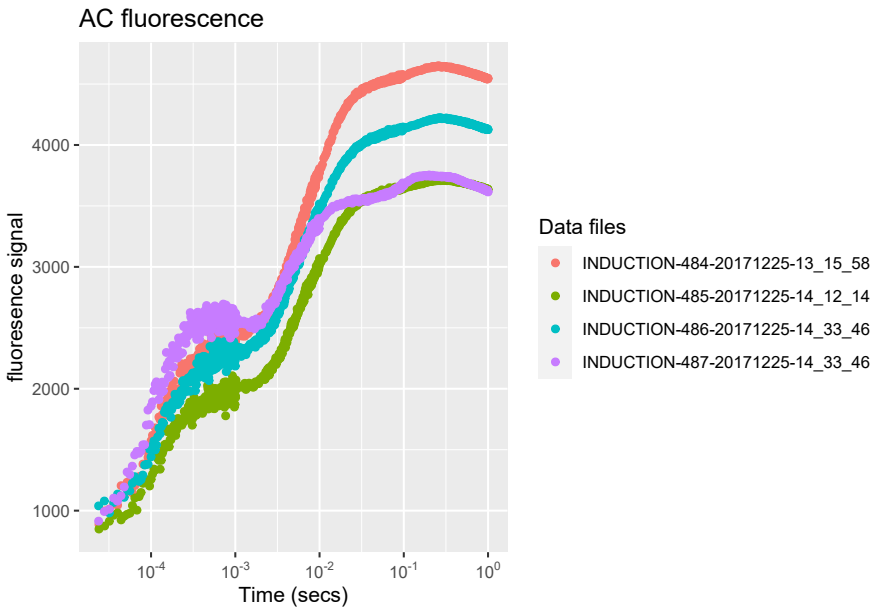


图 10.27: 调制式测量的 ojip 曲线的快速预览

```
jip_dcplot("./data/ojip")
```

ojip 调制式测量光 ojip 曲线的快速预览如图 10.27

ojip 连续式测量光 ojip 曲线的快速预览如图 10.28

建议在开始分析数据前，使用如上方式作图查看数据质量，若使用调制光数据测量的荧光信号太弱，数据点太散，则可以使用连续光测量信号进行分析，对于归一化的荧光参数，二者几乎无差别，当然避免陷入被动的最好方式还是最开始测量时注意检查数据质量

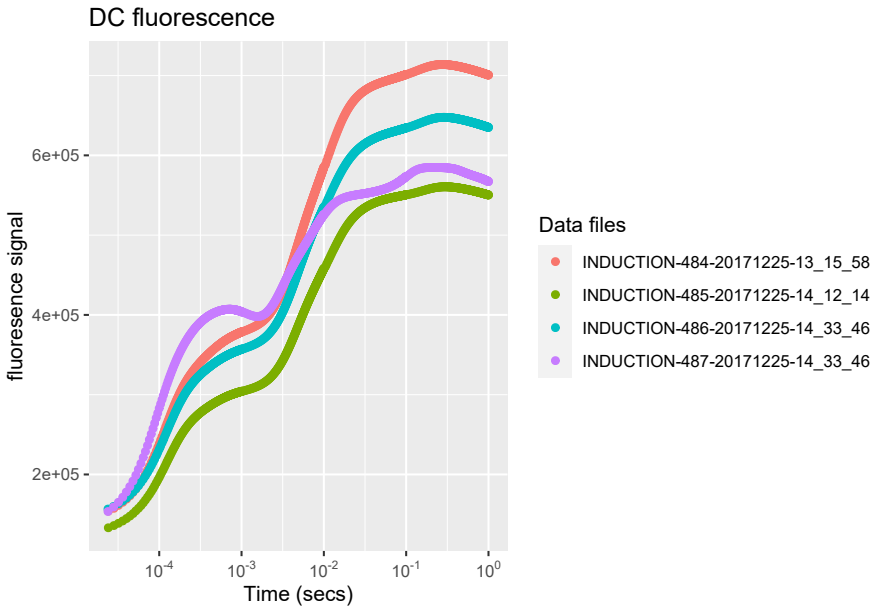


图 10.28: 连续式测量的 ojip 曲线的快速预览

11

大话 *PCA*

PCA 作为一个基础的方法，应用领域覆盖很广，涵盖的统计学相关的内容有均值、标准差、协方差，线性代数中的特征向量及特征值。当然在与光合仪相关的应用方面多数是测量的光合速率、气孔导度等配合植物的其他性状来进行研究，其适用范围，用现在流行的统计学习术语来讲，应属于非监督性学习，即我们并非预测某变量同其预测值的关系，如同我们获得了一系列包含植物光合性状在内的性状，但我们并非将其预测某些结果，如同响应曲线那般，而是用这些形状来分类或者看齐对某目的的重要程度。本章的标题之所以叫大话，是因为这是我结合我自己的理解写的内容，如有错误请谅解，请指出。

另，推荐两个我认为很好的解释 PCA 的链接：

BioTuring's Blog[1]

Principal Component Analysis[2]

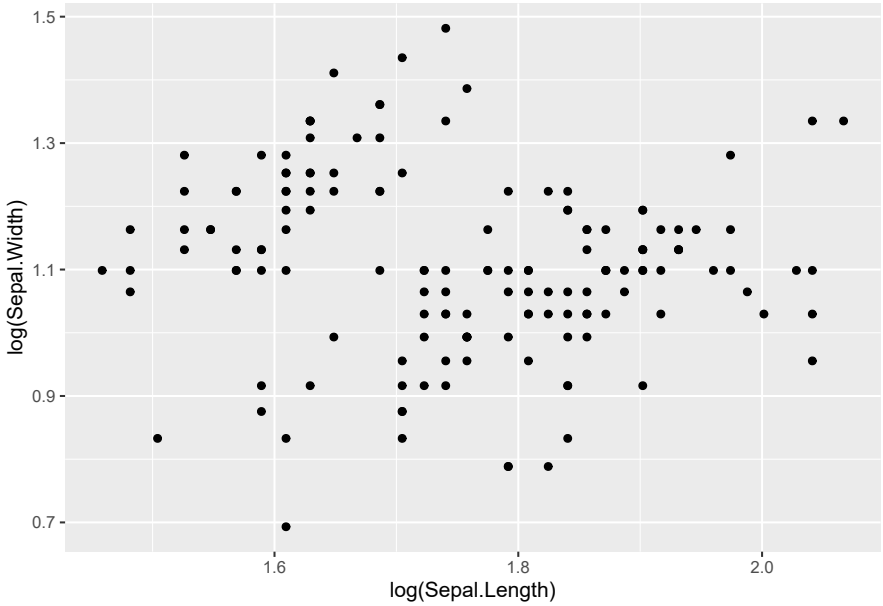
11.1 几何解释

PCA 通常能够反映我们复杂数据集的一些分布特征，例如典型的 `iris` 数据集，我们看到了三个鸢尾种的四个性状：花萼的长宽、花瓣的长宽，如果给我们这么一个数据，我们肉眼很难看出他们的特征来，如果作图也很难表达，因

[1] https://blog.bioturing.com/2018/06/14/principal-component-analysis-explained-simply/

[2] http://setosa.io/ev/principal-component-analysis/

为这是一个四维的数据，假设我们只对前两组也就是花萼的长宽作图，先看一下二维的数据情况：



看上去一言难尽，此时的缺点非常明显，图中的每个点均代表了某个植株的两个性状，这个性状类似于我们说的电子云，杂乱而无规律。这还只是二维数据，如果我们将所有四个轴的数据都显示也就是四维的数据，那么对于我等肉眼凡胎来讲，还是不看为妙。

我在看到上面留的两个网址之前，对主成份分析的疑惑一直没断过，什么是主成份，一直说是主轴旋转，到底是怎么旋转的，我们看一下主成份的 PC1 是怎么来做的:

如图 11.1，假设有三维的数据，可以理解为 iris 数据中前三列的观测值，每个点分别代表了一个植株的前三个性状，PCA 第一步便是中心化，将坐标轴由左边转移到右边的中心点位置作为原点，这就是所谓的平均值中心化的过程，这样就消除了测量时的偏差 [3]，同时一般还要对数据进行缩放，以消除不同单位的影响 [4]，例如有光合速率，有叶片面积的数据，这样能够消除不同单位的影响，通常的缩放例如对数化。

[3] https://www.mathsisfun.com/definitions/bias.html

[4] https://en.wikipedia.org/wiki/Scaling_(geometry)

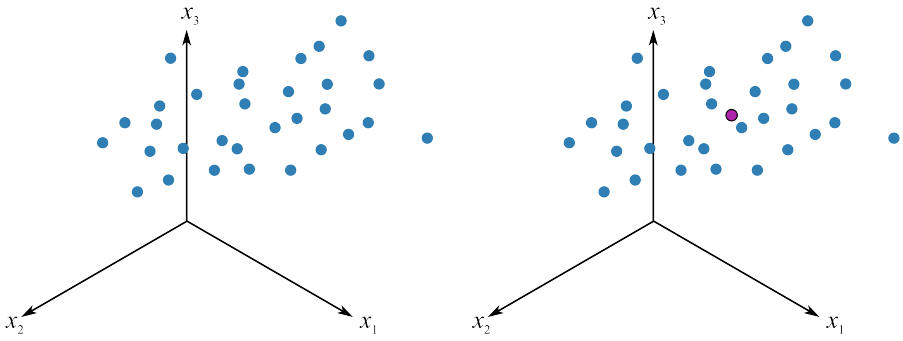


图 11.1: 数据的中心化

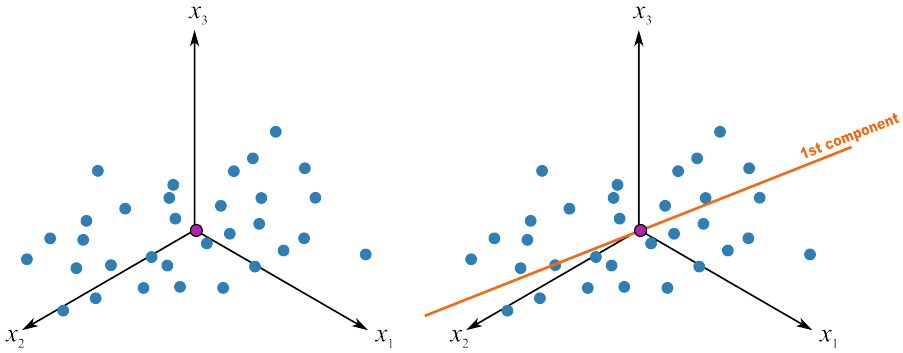


图 11.2: PC1 的诞生

移动坐标系后，因为进行了缩放，所有的数据具有相同的比例。我们这时做一条最佳的拟合线 (图 11.2)，此时发生了两件重要的事情：

- 所有的点投影到这条拟合线上，投影后的点之间的距离是最大的，该方向代表了所有变异最大，也就是最大化的区分所有的数据点，因为我们的目的就是要区分数据点之间的差异。如果在读的各位有跟我一样空间想象力有限的，可以根据图 11.3 的极端情况来理解，蓝色线为最佳拟合线，红色的数据点投影到其上的距离之和当然大于投影到蓝色的拟合线上。

- 所有点和他们相对应的点之间的距离是最小的（也就是残差最小，最佳拟合当然是残差最小），这保证了这些投影的点和原来的数据点尽可能接近，也就是变化后损失的信息最少。

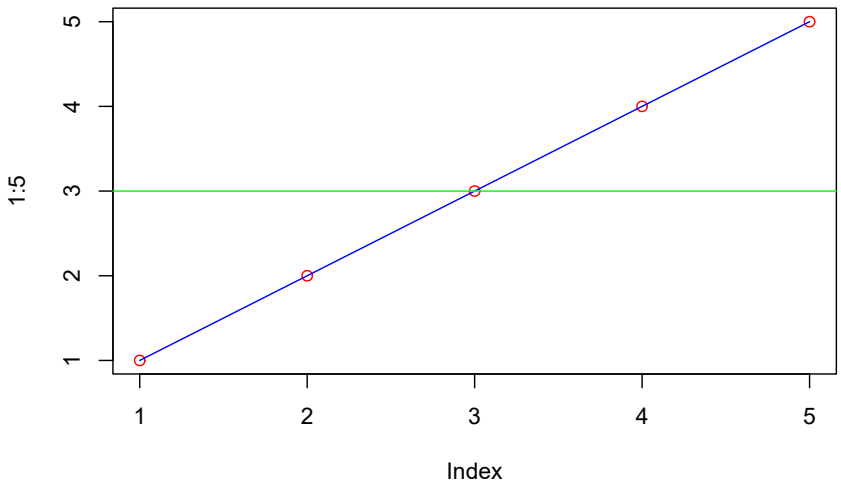


图 11.3: 变异最大的极端情况

以上变化的最终结果为将变异放到最大而误差降低到最小，我们研究的最终目的无非就是这样。这个最佳拟合的线我们将其称之为 PC1，主成份 1。

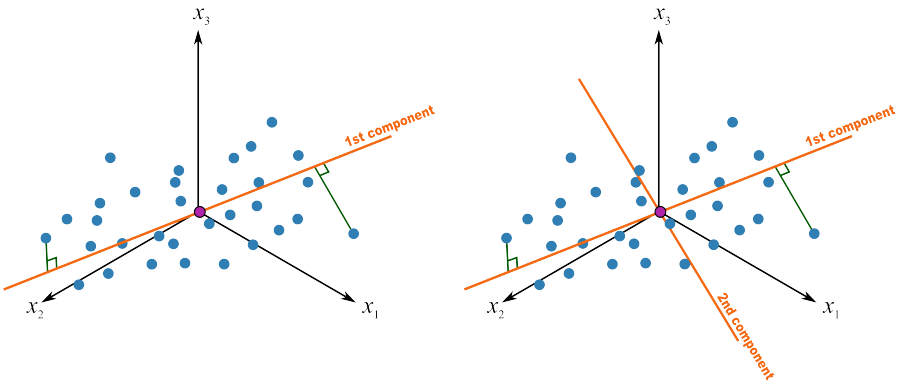


图 11.4: PC2 的诞生

如果我们做一条通过中心点并与 PC1 垂直的线，并不断旋转这条线使其满足：所有投影于其上的点，他们的距离最大，也就是变异最大，这样就做出了第二个主成份，PC2，如图 11.4。**就这样不断的变换，一般情况下，我们可以**

使用 2~3 个主成份来解释绝大部分数据所展示的信息（所有数据都尽可能少的损失信息投影于其上）。

以上为基本变换，如果我们使用 R 计算 PCA (这里推荐两个包 `factoextra` 和 `FactoMineR`)，查看其结果：

```
library("FactoMineR")
library("factoextra")
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve
```

```
ir <- iris[, 1:4]
ir_pca <- PCA(ir, graph = FALSE)

fviz_eig(ir_pca, addlabels = TRUE)
```

我们可以看到，前两个成分可以解释数据 95% 以上的变异，所以我们可以只使用主成份 1 和主成份 2 对我们的结果进行解释。

既然我们使用两个主成份解释整个数据，那么我们怎么把四个性状都放在这个图上呢？这就需要载荷图来展示了，如图 11.6

```
fviz_pca_var(ir_pca)
```

怎么解释呢，因为原始的数据点相当于在坐标轴上，这几个性状分布的产生类似于原始的数据点在拉拽某个性状，影响强的偏向某一方向，换句话说，**这些性状决定主成份时有不同的权重，也就是对主成份影响的权重大小。同时他们是有方向的，如果两个性状方向接近，也就是夹角非常小，说明他们有较强的正相关性，例如图中的花瓣的长宽，如果他们互相垂直，那说明他们无相关性，例如图中花萼的长宽，如果夹角非常大，例如他们几乎在一条直线上，说明具有显著的负相关**。其本质还是所有的单位向量（本例有四个性状，各个性状中心化后形成的轴，也就是向量）在现在的这个二维平面的位置。

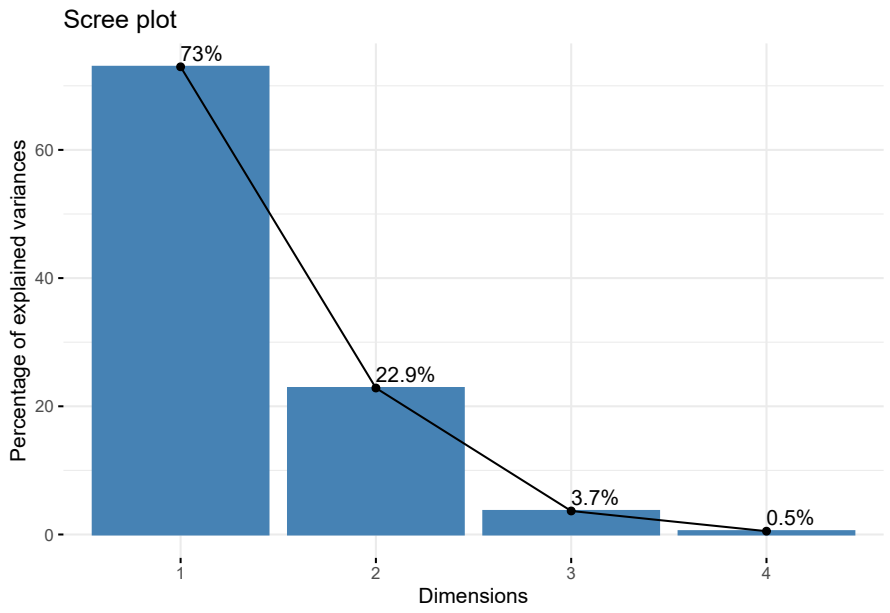


图 **11.5:** iris 碎石图

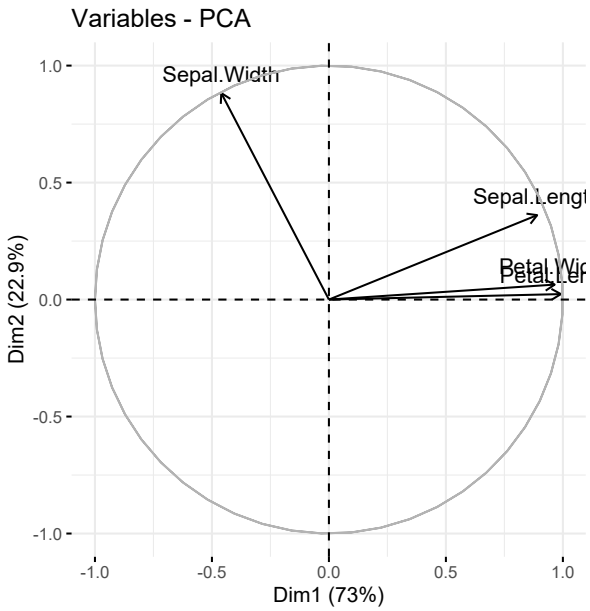


图 **11.6:** iris 载荷图

而如果我想通过这些性状来对鸢尾分类，怎么看呢？那就需要得分图来展示了，如图 11.7 ，得分图本质是各个品种的每个观测值在各个轴上的长度，也就是得分，所以其值都是每个物种性状的加权，每个点代表了每个物种的观测值。

```
fviz_pca_ind(ir_pca, repel = TRUE, col.ind= iris$Species)
```

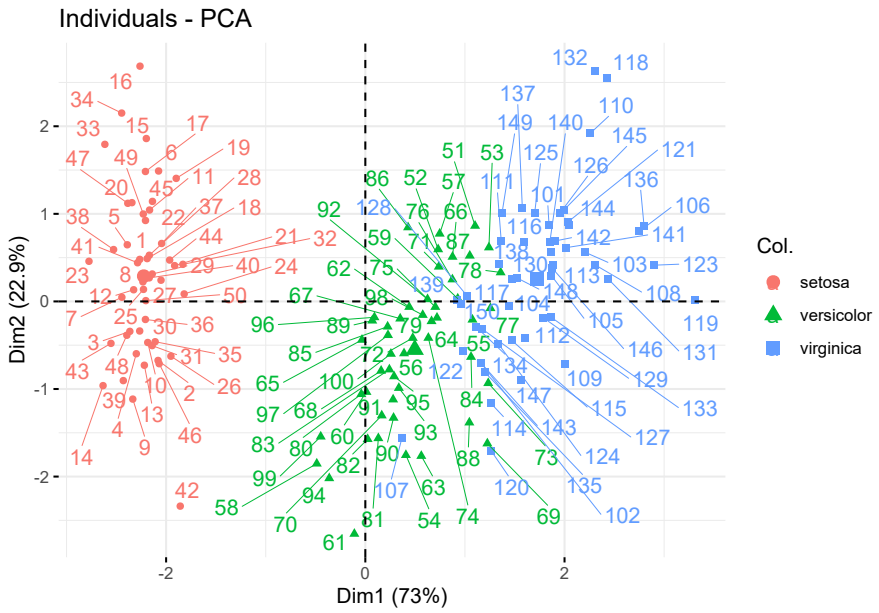


图 11.7: iris 得分图

我们明显看出来，PC1 方向，三个种的鸢尾分成了两大部分，其中 Setosa 延 PC1 同其他两个明显有区别，另外两个在 PC1 上也有较明显的区别，但二者有一定的重合，延 PC2 方向上，三者差别不大。或者反过来讲也可以，品种的差别对 PC1 影响比较显著，这也和我们的碎石图相匹配。

11.2 线性代数解释

11.2.1 特征向量与特征值

特征向量和特征值的数学描述为：

$$A\vec{x} = \lambda\vec{x} \tag{11.1}$$

其中 A 为 n × n 的矩阵，$\vec{x}$ 为非零 n × 1 列向量，λ 是标量，那么 $\vec{x}$ 为 A 的特征向量，λ 为 A 的特征值。

如果仅从数学描述上看，是十分不适合我这样没基础的人来理解的，但我们可以这么理解：

A 为我们观测值（也即我们生态学上测量的数据）组成的 $n \times n$ 矩阵，每行的数据代表了一个观测值，例如光合速率是其中的一行，我们有 n 个，那么 $\vec{x}$ 和 λ 意味着什么呢？再看一眼上面的 (11.1)，这是一个等式，而 $\vec{x}$ 是 n × 1 维度，原来的数据是 n × n 维度，这就是主轴分析是降维分析的意义了，一个一维的特征向量和特征值与我们多维的观测值之间有一个相等的关系。当然，也有很多人在描述时喜欢用坐标轴旋转或这投影来形容的，如果你觉得这样好理解也没问题，不过我脑袋不擅长想象这种多维的空间。这样描述并不十分准确，只是我觉得便于理解，更正确描述应为我们下面所述的例子 (Smith (2002)) ，但降维的原理即来自此处：

$$finaldata = rowfeaturevector \times rowdataajust \tag{11.2}$$

finaldata 很好理解，我们用于 PCA 的最终数据，rowfeature vector 则是我们根据协相关矩阵求得的特征向量的转置，最大的特征向量再最上面，rowdataadjust 则为**原始数据减去标准值**。

因为我手头没有相关的数据，想来想去，我们在 R 里见到最多的与生态学相关的数据也就是 iris，一个关于不同 iris 品种的叶片性状相关的数据，非常生态，我们还是继续使用：

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

数据并不复杂，但是想要得出一些规律性的东西确不那么容易，因为数据量太大了。R 语言里面很多函数可以直接实现 PCA，例如 11.1 内容。这里按照最原始的方法实现一下，加深对原理的理解：

11.2.2 手动实现过程

11.2.2.1 均值的计算

```
iris_mean <- apply(iris[, 1:4], 2, function(x) x - mean(x))
```

我只使用了前四列的性状数据，仅仅是方便结果的展示。

11.2.2.2 计算协方差矩阵

```
iris_cov <- cov(iris_mean)
```

11.2.2.3 计算特征值和特征向量

```
iris_eigen <- eigen(iris_cov)
```

这里就可以看到之前提到的，特征值和特征向量是根据协方差矩阵计算的。我

们选取特征值最大的两个，他们的特征值之和已经占所有的特征值之和相当大的比例：

```
sum(iris_eigen$values[1:2])/sum(iris_eigen$values)
```

```
## [1] 0.9776852
```

也就是说特征值对应的前两项是我们最终降维所需要的，即最终的两个主轴为 150×2 维矩阵，即我们原来的 150×4 的矩阵乘以我们选取的前两个特征值对应的 4×2 组成的矩阵。

实际应用中我们不需要这么做了，因为太浪费时间了，我们直接用函数来得到结果即可。

11.2.3 prcomp 的实现

我们用 prcomp 来简化实现过程，注意，根据 Kemp (2003)，我们把 iris 数据对数化一下，并提取所有物种名字：

```
data("iris")
iris_pca <- log(iris[, 1:4])
iris_species <- iris$Species
```

需要注意，我们处理时使用特征中心化。即每一维的数据都减去该维的均值。这里的“维”指的就是一个特征（或属性），变换之后每一维的均值都变成了 0 (参考 (11.2))。

```
value_pca <- prcomp(iris_pca, center = TRUE, scale. = TRUE)
## 查看结果
summary(value_pca)
```

```
## Importance of components:
##                            PC1    PC2     PC3     PC4
## Standard deviation     1.7125 0.9524 0.36470 0.16568
## Proportion of Variance 0.7331 0.2268 0.03325 0.00686
## Cumulative Proportion  0.7331 0.9599 0.99314 1.00000
```

可以看到结果同我们最开始的计算相似，不同的是我们进行了对数化，而且根据 Kemp (2003)，这个结果更合适。

12

环境与配置

本文的内容所完成的 sessioninfo 如下：

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19041)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Chinese (Simplified)_China.936
## [2] LC_CTYPE=Chinese (Simplified)_China.936
## [3] LC_MONETARY=Chinese (Simplified)_China.936
## [4] LC_NUMERIC=C
## [5] LC_TIME=Chinese (Simplified)_China.936
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] factoextra_1.0.7   FactoMineR_2.3     jiptest_0.1.3      gridExtra_2.3
##  [5] plyr_1.8.6         latex2exp_0.4.0    RColorBrewer_1.1-2 racir_1.0.0
##  [9] nls2_0.2           proto_1.0.0        purrr_0.3.4        ggplot2_3.3.2
```

```
## [13] minpack.lm_1.2-1   plantecophys_1.4-4 readphoto_0.0.1
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-150         tools_4.0.3          backports_1.2.0
##  [4] R6_2.5.0             mgcv_1.8-33          colorspace_1.4-1
##  [7] MESS_0.5.7           withr_2.3.0          tidyselect_1.1.0
## [10] rematch_1.0.1        curl_4.3             compiler_4.0.3
## [13] XLConnect_1.0.1      flashClust_1.01-2    labeling_0.4.2
## [16] bookdown_0.21        mosaicCore_0.8.0     scales_1.1.1
## [19] readr_1.4.0          stringr_1.4.0        digest_0.6.27
## [22] ggformula_0.9.4      foreign_0.8-80       rmarkdown_2.5
## [25] rio_0.5.16           pkgconfig_2.0.3      htmltools_0.5.0
## [28] geeM_0.10.1          highr_0.8            rlang_0.4.8
## [31] readxl_1.3.1         rstudioapi_0.11      farver_2.0.3
## [34] generics_0.1.0       zip_2.1.1            dplyr_1.0.2
## [37] car_3.0-10           magrittr_1.5         leaps_3.1
## [40] Matrix_1.2-18        Rcpp_1.0.5           munsell_0.5.0
## [43] abind_1.4-5          geepack_1.3-1        lifecycle_0.2.0
## [46] scatterplot3d_0.3-41 stringi_1.5.3        yaml_2.2.1
## [49] carData_3.0-4        MASS_7.3-53          ggstance_0.3.4
## [52] grid_4.0.3           ggrepel_0.8.2        forcats_0.5.0
## [55] crayon_1.3.4         lattice_0.20-41      haven_2.3.1
## [58] splines_4.0.3        hms_0.5.3            knitr_1.30
## [61] pillar_1.4.6         ggpubr_0.4.0         ggsignif_0.6.0
## [64] glue_1.4.1           evaluate_0.14        data.table_1.13.2
## [67] png_0.1-7            vctrs_0.3.4          tweenr_1.0.1
## [70] cellranger_1.1.0     gtable_0.3.0         polyclip_1.10-0
## [73] tidyr_1.1.2          openxlsx_4.2.3       xfun_0.19
## [76] ggforce_0.3.2        broom_0.7.2          rstatix_0.6.0
## [79] tibble_3.0.4         rJava_0.9-13         cluster_2.1.0
## [82] ellipsis_0.3.1
```

参考文献

Aganchich, B., Wahbi, S., Loreto, F., and Centritto, M. (2009). Partial root zone drying: regulation of photosynthetic limitations and antioxidant enzymatic activities in young olive (olea europaea) saplings. *Tree Physiology*, 29(5):685.

Ball, J. T., Woodrow, I. E., and Berry, J. A. (1987). *A Model Predicting Stomatal Conductance and its Contribution to the Control of Photosynthesis under Different Environmental Conditions*. Springer Netherlands.

Baly, E. (1935). The kinetics of photosynthesis. *Proceedings of the Royal Society of London Series B (Biological Sciences)*, (117):218–239.

Bouvier, A. and Huet, S. (1994). nls2 nonlinear regression by s plus functions. *Computational Statistics & Data Analysis*, 18(1):187–190.

Buckley, T. N., Martorell, S., Diazespejo, A., Tomàs, M., and Medrano, H. (2014). Is stomatal conductance optimized over both time and space in plant crowns? a field test in grapevine vitis vinifera. *Plant Cell & Environment*, 37(12):2707.

Caemmerer, S. V. (2000). Biochemical models of leaf photosynthesis. *Quarterly Review of Biology*, page 165.

Centritto, M., Loreto, F., and Chartzoulakis, K. (2003). The use of low co2 to estimate diffusional and non-diffusional limitations of photosynthetic capacity of salt-stressed olive saplings. *Plant Cell and Environment*, 26(4):585–594.

Coursolle, C., Prud' homme, G. O., Lamothe1, M., and Isabel1, N. (2019).

Measuring rapid a–ci curves in boreal conifers: Black spruce and balsam fir. *Frontiers in Plant Science*.

Cowan, I. R. and Farquhar, G. D. (1977). Stomatal function in relation to leaf metabolism and environment. *Symposia of the Society for Experimental Biology*, 31(23):471.

De Kauwe, M. G., Lin, Y. S., Wright, I. J., Medlyn, B. E., Crous, K. Y., Ellsworth, D. S., Maire, V., Prentice, I. C., Atkin, O. K., and Rogers, A. (2016). A test of the one point method for estimating maximum carboxlation capacity from field measured, light saturated photosynthesis. *New Phytologist*, 210(3):1130.

Duursma, R. A. (2015). Plantecophys-an r package for analysing and modelling leaf gas exchange data. *Plos One*, 10(11):e0143346.

Duursma, R. A. and Medlyn, B. E. (2012). Maespa: a model to study interactions between water limitation, environmental drivers and vegetation function at tree and stand levels, with an example application to co_2 drought interactions. *Geoscientific Model Development Discussions*, 5(4):919–940.

Duursma, R. A., Payton, P., Bange, M. P., Broughton, K. J., Smith, R. A., Medlyn, B. E., and Tissue, D. T. (2013). Near-optimal response of instantaneous transpiration efficiency to vapour pressure deficit, temperature and co_2 in cotton (gossypium hirsutum l.). *Agricultural & Forest Meteorology*, 168(1):168–176.

Ellsworth, D. S., Crous, K. Y., Lambers, H., and Cooke, J. (2015). Phosphorus recycling in photorespiration maintains high photosynthetic capacity in woody species. *Plant Cell and Environment*, 38(6):1142–1156.

Elzhov, T. V., Mullen, K. M., Spiess, A. N., and Bolker, B. (2016). minpack.lm: R interface to the levenberg-marquardt nonlinear least-squares algorithm found in minpack, plus support for bounds.

Farquhar, G. D., Caemmerer, S. V., and Berry, J. A. (1980). A biochemical

model of photosynthetic co_2 assimilation in leaves of c3 species. *Planta*, 149(1):78–90.

Flexas J, A. A. and A, B. J. (2007). Analysis of leakage in irga's leaf chambers of open gas exchange systems: quantification and its effects in photosynthesis parameterization. *Journal of Experimental Botany*, 58(6):1533.

Gu, L., Pallardy, S. G., Tu, K., Law, B. E., and Wullschleger, S. D. (2010). Reliable estimation of biochemical parameters from c-3 leaf photosynthesis-intercellular carbon dioxide response curves. *Plant Cell & Environment*, 33(11):1852–1874.

Haworth, M., Killi, D., Materassi, A., and Raschi, A. (2015). Coordination of stomatal physiological behavior and morphology with carbon dioxide determines stomatal control. *American Journal of Botany*, 102(5):677–688.

Haworth, M., Marino, G., and Centritto, M. (2018). An introductory guide to gas exchange analysis of photosynthesis and its application to plant phenotyping and precision irrigation to enhance water use efficiency. *Journal of Water and Climate Change*.

Heath, J. R., Ayres, E., Possell, M., Bardgett, R. D., Black, H. I. J., Grant, H., Ineson, P., and Kerstiens, G. (2005). Rising atmospheric co2 reduces sequestration of root-derived soil carbon. *Science*, 309(5741):1711–1713.

Hikosaka, K., Ishikawa, K., Borjigidai, A., Muller, O., and Onoda, Y. (2006). Temperature acclimation of photosynthesis: mechanisms involved in the changes in temperature dependence of photosynthetic rate. *Journal of Experimental Botany*, 57(2):291.

Huang, H., Dou, X., Sun, P., Deng, B., Wu, G., and Peng, C. (2009). Comparison of photosynthetic characteristics in two ecotypes of jatropha curcas in summer. *Acta Ecologica Sinica*, (29):2861–2867.

Jones, H. G. (1993). *Plants and microclimate. A quantitative approach to environmental plant physiology.* Cambridge University Press.

Joseph R., S., Rachael K., A., and David T., H. (2019). Using multirate rapid aci curves as a tool to explore new questions in the photosynthetic physiology of plants. *New Phytologist*.

Kemp, F. (2003). Modern applied statistics with s. *Journal of The Royal Statistical Society Series D-the Statistician*, 52(4):704–705.

Lauteri, M., Haworth, M., Serraj, R., Monteverdi, M. C., and Centritto, M. (2014). Photosynthetic diffusional constraints affect yield in drought stressed rice cultivars during flowering. *PLOS ONE*, 9(10).

Leuning, R. (1995). A critical appraisal of a combined stomatal-photosynthesis model for c3 plants. *Plant Cell and Environment*, 18(4):339–355.

Lobo, F. D. A., Barros, M. P. D., Dalmagro, H. J., Dalmolin, . C., Pereira, W. E., de Souza, . C., Vourlitis, G. L., and Ortíz, C. E. R. (2013). Fitting net photosynthetic light-response curves with microsoft excel a critical look at the models. *Photosynthetica*, 51(3):445–456.

Long, S. P. and Bernacchi, C. J. (2003). Gas exchange measurements, what can they tell us about the underlying limitations to photosynthesis? procedures and sources of error. *Journal of Experimental Botany*, 54(392):2393.

Marshall, B. and Biscoe, P. V. (1980). A model for c3 leaves describing the dependence of net photosynthesis on irradiance. *Journal of Experimental Botany*, 31(1):29–39.

Medlyn, B. E., Dreyer, E., Ellsworth, D., Forstreuter, M., Harley, P. C., Kirschbaum, M. U. F., Roux, X. L., Montpied, P., Strassemeyer, J., and Walcroft, A. (2002). Temperature response of parameters of a biochemically based model of photosynthesis. ii. a review of experimental data review. *Plant Cell & Environment*, 25(9):1167–1179.

Medlyn, B. E., Duursma, R. A., Eamus, D., Ellsworth, D. S., Prentice, I. C., Barton, C. V. M., Crous, K. Y., Angelis, P. D., Freeman, M., and

Wingate, L. (2011). Reconciling the optimal and empirical approaches to modelling stomatal conductance. *Global Change Biology*, 17(6):2134–2144.

Prado, C. H. and Moraes, J. A. P. V. D. (1997). Photosynthetic capacity and specific leaf mass in twenty woody species of cerrado vegetation under field conditions. *Photosynthetica*, 33(1):103–112.

Smith, L. I. (2002). A tutorial on principal component analysis.

Stinziano, J. R., Mcdermitt, D. K., Lynch, D. J., Saathoff, A. J., Morgan, P. B., and Hanson, D. T. (2018). The rapid a/ci response: a guide to best practices. *New Phytologist*.

Stinziano, J. R., Morgan, P. B., Lynch, D. J., Saathoff, A. J., Mcdermitt, D. K., and Hanson, D. T. (2017). The rapid a-ci response: photosynthesis in the phenomic era. *Plant Cell & Environment*, 40.

Thornley, J. H. M. (1976). Mathematical models in plant physiology. *London: Academic Press*.

Tuzet, A., Perrier, A., and Leuning, R. (2003). A coupled model of stomatal conductance, photosynthesis and transpiration. *Plant Cell and Environment*, 26(7):1097–1116.

Wullschleger, S. D. (1993). Biochemical limitations to carbon assimilation in c3 plants—a retrospective analysis of the a-ci curves from 109 species. *Journal of Experimental Botany*, 44(262):907–920.

Yin, X., Struik, P. C., Romero, P., Harbinson, J., Evers, J. B., Der Putten, P. E. L. V., and Vos, J. (2009). Using combined measurements of gas exchange and chlorophyll fluorescence to estimate parameters of a biochemical c3 photosynthesis model: a critical appraisal and a new integrated approach applied to leaves in a wheat (triticum aestivum) canopy. *Plant Cell and Environment*, 32(5):448–464.

Zhang, X., Shen, S., and Song, J. (2009). The vertical distribution of

cotton leaf nitrogen content and photosynthetic characteristics in the north china plain. *Acta Ecologica Sinica*, (29):1893–1898.

ZiPiao, Y. (2010). A review on modeling of responses of photosynthesis to light and co_2. *Chinese Journal of Plant Ecology*, (06).