8. S-PLUS Interface

The S-PLUS custom interface, *libsdi*, provides interactive access to a database from the S-PLUS prompt.

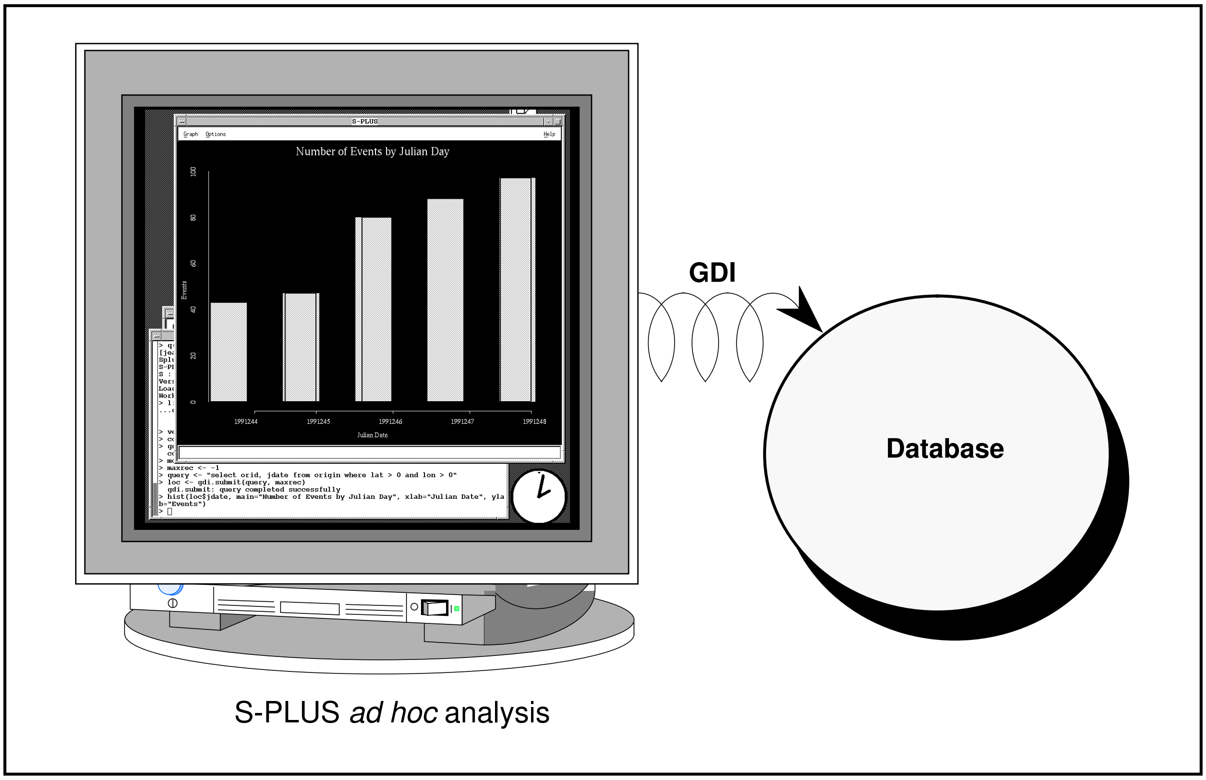

FIGURE S-1. S-PLUS Custom Interface

The GDI interface complements Data Junction, a data conversion toolkit marketed by Statistical Sciences, Inc. Data Junction is a stand alone program that converts data between S-PLUS and other applications using a “hub and spoke” architecture and intermediate files. It is primarily a data mover to extract and import data.

The GDI plugs S-PLUS right into the target database to support concurrent interactive database querying and S-PLUS analysis. Data transfer from the database to S-PLUS is transparent to the user. All ANSI SQL commands are available from inside S-PLUS.

To use the S-PLUS interface, the user must know:

- The SQL query language.
- The S Language.
- How to use the functions described in this section.

8.1 Starting S-PLUS

Figure S-2 shows how to start S-PLUS and load the *libsdi* interface:

```
xterm
[jean]angrboda.css.gov% Splus
S-PLUS : Copyright (c) 1988, 1992 Statistical Sciences, Inc.
S : Copyright AT&T.
Version 3.1 Release 1 for Sun SPARC, SunOS 4.x : 1992
Load New Splus Database Interface (V2.0) by typing libsdi()
Working data will be in .Data
> libsdi()
...dynamically loading New Splus Database Interface (V2.0)
                   type "library(help="libsdi")" for help...

> 
```

FIGURE S-2. Starting S-PLUS

On-line help is available by entering *library(help=”libsdi”).*

8.2 Connecting to a Database

gdi.open establishes a connection to the database and takes the following parameters:

vendor	name of the database product (oracle, sybase, ingres)
connect	login string for the target database

Figure S-3 shows a user connecting to an ORACLE database using an ORACLE Version 6 sql*-net connect string.

```
xterm
[jean]angrboda.css.gov% Splus
S-PLUS : Copyright (c) 1988, 1992 Statistical Sciences, Inc.
S : Copyright AT&T.
Version 3.1 Release 1 for Sun SPARC, SunOS 4.x : 1992
Load New Splus Database Interface (V2.0) by typing libsdi()
Working data will be in .Data
> libsdi()
...dynamically loading New Splus Database Interface (V2.0)
                    type "library(help="libsdi")" for help...

> vendor <- "oracle"
> connect <- "geodemo/geodemo@t:sleipnir:dev"
> gdi.open(vendor, connect)
  completed successfully
> 
```

FIGURE S-3. Connecting to a Database

If the database connect fails, Figure S-4 shows how the database error is reported.

```
xterm
[jean]angrboda.css.gov% Splus
S-PLUS : Copyright (c) 1988, 1992 Statistical Sciences, Inc.
S : Copyright AT&T.
Version 3.1 Release 1 for Sun SPARC, SunOS 4.x : 1992
Load New Splus Database Interface (V2.0) by typing libsdi()
Working data will be in .Data
> libsdi()
...dynamically loading New Splus Database Interface (V2.0)
                    type "library(help="libsdi")" for help...

> vendor <- "oracle"
> connect <- "bad_login/badd_passwd@t:sleipnir:dev"
> gdi.open(vendor, connect)
  gdi_open4s: Error -1017: 'ORA-01017: invalid username/password; logon denied
'
  ERROR opening database
> 
```

FIGURE S-4. Bad Database Connection

Although the GDI supports multiple concurrent connections, the S-PLUS interface currently supports only one active database connection at a time.

8.3 Executing SQL Queries

gdi.submit executes SQL queries and takes the following parameters:

query	String containing a complete SQL query. Oracle and Ingres queries should not have a semicolon at the end. A Sybase query string should contain only one query.
maxrecs	Maximum number of records to fetch. If set to -1, all records will be returned. If set to 0, up to 500 records will be returned. Otherwise set it to the maximum number of records you want.

Figure S-5 shows how to build and execute a dynamic SQL query.

```
> maxrec <- -1
> query <- "select * from origin where lat > 0 and lon > 0"
> loc <- gdi.submit(query, maxrec)
  gdi.submit: query completed successfully
> loc$tuple.count
[1] 862
> attributes(loc)
$names:
 [1] "lat"         "lon"         "depth"       "time"        "orid"
 [6] "evid"        "jdate"       "nass"        "ndef"        "ndp"
[11] "grn"         "srn"         "etype"       "depdp"       "dtype"
[16] "mb"          "mbid"        "ms"          "msid"        "ml"
[21] "mlid"        "algorithm"   "auth"        "commid"      "lddate"
[26] "tuple.count"

>
```

FIGURE S-5. Executing a SQL Query

Figure S-6 executes a query containing a join.

```
> query <- "select a.jdate, a.lat, a.lon, b.smajax from origin a, origerr b wher
e a.orid=b.orid and a.lat > 0 and a.lon > 0"
> loc <- gdi.submit(query, maxrec)
  gdi.submit: query completed successfully
> loc$tuple.count
[1] 862
> attributes(loc)
$names:
[1] "jdate"       "lat"         "lon"         "smajax"      "tuple.count"

>
```

FIGURE S-6. Executing a JOIN Query

Any valid SQL query may be executed. However, it is important to understand that the GDI passes queries straight through to the target database. So a query containing the Oracle outer join operator will produce an error if sent to a Sybase database and *vice versus*.

8.4 Plotting Results

The commands in Figure S-7 create a motif window, execute an SQL query, and produce a histogram in the motif window using the results from the SQL query.

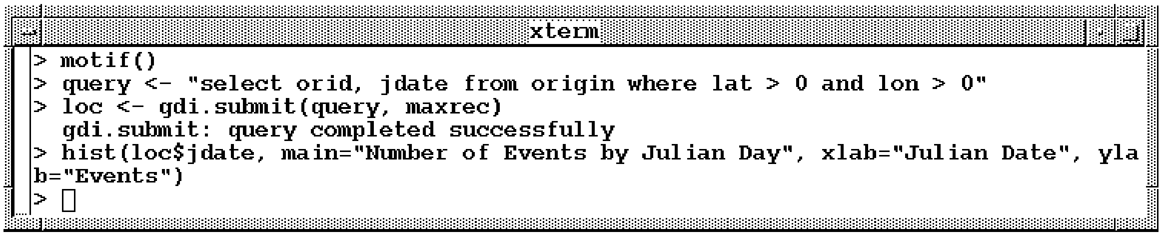

FIGURE S-7. Plotting Results

Figure S-8 shows the results in the motif window.

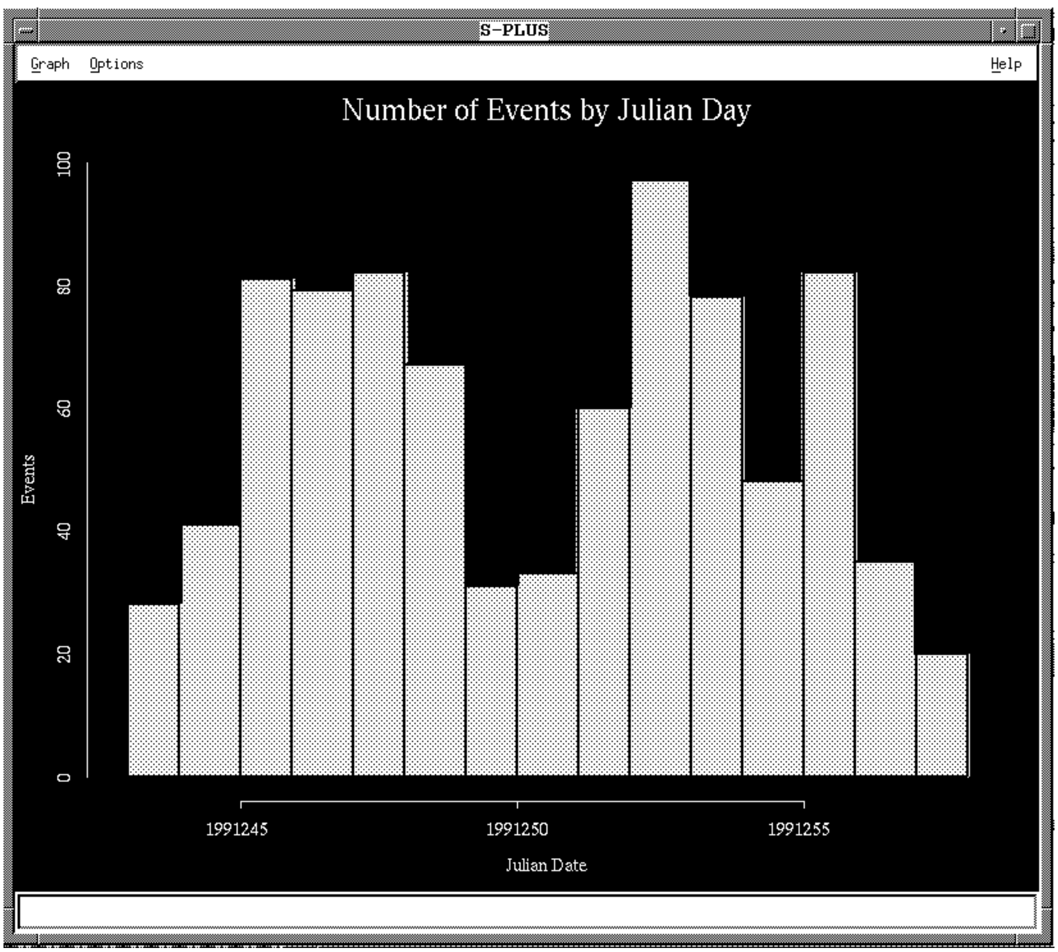

FIGURE S-8. S-PLUS Histogram

8.5 Transaction Management

If queries executed by *gdi.submit* change the database, those changes do not become permanent until a *commit* occurs. A *commit* makes all changes permanent. A *rollback* undoes all changes.

Two conditions automatically cause a commit:

- A DDL statement, such as create or drop, commits pending changes even if the statement itself fails.
- *gdi.close*, described in Section 8.6, commits pending changes before terminating the database connection.

In general, it is better to explicitly commit or rollback by storing the *commit* or *rollback* statement in a query string and executing it with *gdi.submit*.

8.6 Exiting S-PLUS

gdi.close disconnects from the database. The commands in Figure S-9 disconnect from the database and exit S-PLUS.

```
xterm
  gdi.submit: query completed successfully
> hist(loc$jdate, main="Number of Events by Julian Day", xlab="Julian Date", yla
b="Events")
> gdi.close()
  database closed successfully
> q()
[jean]angrboda.css.gov%
```

FIGURE S-9. Exiting S-PLUS

8.7 Summary of S-PLUS Commands

Currently S-PLUS interface functions, summarized in Table 13., support a subset of the GDI. Additional functions will be added as GDI users request them.

Table 13. Summary of S-PLUS Interface Functions

S-PLUS Function	*Description*	*GDI Man Page*
gdi.open	Establishes a connection to the database.	gdi_open(3)
gdi.close	Closes a connection to the database.	gdi_close(3)
gdi.submit	Executes an SQL query.	gdi_submit(3)